

## IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of

Junji TAKAGI

Serial No. (unknown)

09/982,950

Filed herewith

STUB SEARCH LOADING SYSTEM  
AND METHOD, SERVER APPARATUS,  
CLIENT APPARATUS, AND  
COMPUTER-READABLE RECORDING  
MEDIUM



**CLAIM FOR FOREIGN PRIORITY UNDER 35 U.S.C. 119  
AND SUBMISSION OF PRIORITY DOCUMENT**

Assistant Commissioner for Patents

Washington, D.C. 20231

Sir:

Attached hereto is a certified copy of applicant's  
corresponding patent application filed in Japan, on 23 October  
2000, under No. 322269/2000.

Applicant herewith claims the benefit of the  
priority filing date of the above-identified application for  
the above-entitled U.S. application under the provisions of 35  
U.S.C. 119.

Respectfully submitted,

YOUNG &amp; THOMPSON

By

A handwritten signature in cursive script, reading "Benoît Castel".

Benoît Castel  
Attorney for Applicant  
Registration No. 35,041  
Customer No. 00466  
745 South 23rd Street  
Arlington, VA 22202  
Telephone: 703/521-2297

October 22, 2001

日本国特許庁  
JAPAN PATENT OFFICE



別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office

出願年月日

Date of Application:

2000年10月23日

出願番号

Application Number:

特願2000-322269

出願人  
Applicant(s):

日本電気株式会社

CERTIFIED COPY OF  
PRIORITY DOCUMENT

2001年 8月31日

特許庁長官  
Commissioner,  
Japan Patent Office

及川耕造

出証番号 出証特2001-3077548

【書類名】 特許願

【整理番号】 49260001

【提出日】 平成12年10月23日

【あて先】 特許庁長官殿

【国際特許分類】 G06F 9/46

【発明者】

【住所又は居所】 東京都港区芝五丁目7番1号 日本電気株式会社内

【氏名】 高木 淳司

【特許出願人】

【識別番号】 000004237

【氏名又は名称】 日本電気株式会社

【代理人】

【識別番号】 100088959

【弁理士】

【氏名又は名称】 境 廣巳

【手数料の表示】

【予納台帳番号】 009715

【納付金額】 21,000円

【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【包括委任状番号】 9002136

【プルーフの要否】 要

【書類名】 明細書

【発明の名称】 スタブ検索ローディングシステム及び方法、サーバ装置、クライアント装置並びにコンピュータ可読記録媒体

【特許請求の範囲】

【請求項1】 クライアントからサーバにリモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするスタブ検索ローディングシステムであって、

クライアント側に、スタブ名とクライアント識別子とを指定したスタブ要求をサーバに送出して、その応答として返されるスタブを受信するスタブ検索部を備え、

サーバ側に、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合と、クライアントからのスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ集合の中から適切なスタブを検索して要求元のクライアントに返すスタブ検索インターフェースとを備えたスタブ検索ローディングシステム。

【請求項2】 クライアントからサーバにリモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするスタブ検索ローディングシステムであって、

クライアント側に、スタブ名とクライアント識別子とを指定したスタブ要求をサーバに送出して、その応答として返されるスタブを受信するスタブ検索部を備え、

サーバ側に、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に生成するスタブ生成部と、クライアントからのスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ生成部に要求元のクライアントの実行環境に適したスタブを生成させ、生成されたスタブを要求元のクライアントに返すスタブ検索インターフェースとを備えたスタブ検索ローディングシステム。

【請求項 3】 クライアントからサーバにリモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするスタブ検索ローディングシステムであって、

クライアント側に、スタブ名とクライアント識別子とを指定したスタブ要求をサーバに送出して、その応答として返されるスタブを受信するスタブ検索部を備え、

サーバ側に、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合と、前記スケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に生成するスタブ生成部と、クライアントからのスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ集合の中から適切なスタブを検索して要求元のクライアントに返し、適切なスタブが前記スタブ集合に存在しなければ指定されたスタブ名とクライアント識別子に基づいて前記スタブ生成部に要求元のクライアントの実行環境に適したスタブを生成させ、生成されたスタブを要求元のクライアントに返すスタブ検索インターフェースとを備えたスタブ検索ローディングシステム。

【請求項 4】 クライアントからサーバにリモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするスタブ検索ローディング方法であって、

(a) クライアントからサーバに、スタブ名とクライアント識別子とを指定したスタブ要求を送出するステップ、

(b) サーバにおいて、指定されたクライアント識別子に基づいて、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合の中から 1 つのスタブ集合を選択するステップ、

(c) サーバにおいて、指定されたスタブ名のスタブを前記選択したスタブ集合の中から選択するステップ、

(d) サーバからクライアントに、前記選択されたスタブを送出するステップ、

(e) クライアントにおいて、サーバから送出された前記スタブを受信するステ

ップを含むスタブ検索ローディング方法。

【請求項5】 クライアントからサーバにリモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするスタブ検索ローディング方法であって、

(a) クライアントからサーバに、スタブ名とクライアント識別子とを指定したスタブ要求を送出するステップ、

(b) サーバにおいて、指定されたスタブ名とクライアント識別子に基づいて、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを生成するステップ、

(c) サーバからクライアントに、前記生成したスタブを送出するステップ、

(d) クライアントにおいて、サーバから送付された前記スタブを受信するステップを含むスタブ検索ローディング方法。

【請求項6】 クライアントからサーバにリモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするスタブ検索ローディング方法であって、

(a) クライアントからサーバに、スタブ名とクライアント識別子とを指定したスタブ要求を送出するステップ、

(b) サーバにおいて、指定されたクライアント識別子に基づいて、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合の中から1つのスタブ集合を選択し、且つ、該選択したスタブ集合の中から指定されたスタブ名のスタブを検索するステップ、

(c) サーバにおいて、ステップbの検索に失敗した場合に、指定されたスタブ名とクライアント識別子に基づいて、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを生成するステップ、

(d) サーバからクライアントに、ステップbの検索に成功した場合にはステップbで検索されたスタブを、ステップbの検索に失敗した場合にはステップcで生成されたスタブを送出するステップ、

(e) クライアントにおいて、サーバから送付された前記スタブを受信するステ

ップを含むスタブ検索ローディング方法。

【請求項 7】 リモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをクライアントからのスタブ要求に応じて要求元のクライアントに提供するサーバ装置であって、

クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合と、クライアントからのスタブ名とクライアント識別子とを指定したスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ集合の中から適切なスタブを検索して要求元のクライアントに返すスタブ検索インターフェースとを備えたサーバ装置。

【請求項 8】 リモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをクライアントからのスタブ要求に応じて要求元のクライアントに提供するサーバ装置であって、

クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に生成するスタブ生成部と、クライアントからのスタブ名とクライアント識別子とを指定したスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ生成部に要求元のクライアントの実行環境に適したスタブを生成させ、生成されたスタブを要求元のクライアントに返すスタブ検索インターフェースとを備えたサーバ装置。

【請求項 9】 リモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをクライアントからのスタブ要求に応じて要求元のクライアントに提供するサーバ装置であって、

クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合と、前記スケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に生成するスタブ生成部と、クライアントからのスタブ名とクライアント識別子とを指定したスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ集合の中から適切なスタブを検索して要求元のクライア

ントに返し、適切なスタブが前記スタブ集合に存在しなければ指定されたスタブ名とクライアント識別子に基づいて前記スタブ生成部に要求元のクライアントの実行環境に適したスタブを生成させ、生成されたスタブを要求元のクライアントに返すスタブ検索インターフェースとを備えたサーバ装置。

【請求項 1 0】 リモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするクライアント装置であって、

クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合、及びクライアントからのスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ集合の中から適切なスタブを検索して要求元のクライアントに返すスタブ検索インターフェースとを備えたサーバ装置に対して、スタブ名とクライアント識別子とを指定したスタブ要求を送出し、その応答として返されるスタブを受信するスタブ検索部を備えたクライアント装置。

【請求項 1 1】 リモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするクライアント装置であって、

クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に生成するスタブ生成部及び、クライアントからのスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ生成部に要求元のクライアントの実行環境に適したスタブを生成させ、生成されたスタブを要求元のクライアントに返すスタブ検索インターフェースとを備えたサーバ装置に対して、スタブ名とクライアント識別子とを指定したスタブ要求をサーバに送出し、その応答として返されるスタブを受信するスタブ検索部を備えたクライアント装置。

【請求項 1 2】 リモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするクライアント装置であって、

クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合、前記スケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に生成するスタブ生成部及び、クライアントからのスタブ要求時、指定



されたスタブ名とクライアント識別子に基づいて前記スタブ集合の中から適切なスタブを検索して要求元のクライアントに返し、適切なスタブが前記スタブ集合に存在しなければ指定されたスタブ名とクライアント識別子に基づいて前記スタブ生成部に要求元のクライアントの実行環境に適したスタブを生成させ、生成されたスタブを要求元のクライアントに返すスタブ検索インターフェースとを備えたサーバ装置に対して、スタブ名とクライアント識別子とを指定したスタブ要求をサーバに送出し、その応答として返されるスタブを受信するスタブ検索部を備えたクライアント装置。

【請求項13】 クライアントからサーバにリモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするスタブ検索ローディングプログラムを記録したコンピュータ可読記録媒体であって、

クライアントを構成するコンピュータを、スタブ名とクライアント識別子とを指定したスタブ要求をサーバに送出して、その応答として返されるスタブを受信するスタブ検索手段として機能させ、

サーバを構成するコンピュータを、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合の記憶手段、クライアントからのスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ集合の中から適切なスタブを検索して要求元のクライアントに返すスタブ検索インターフェース手段として機能させるプログラムを記録したコンピュータ可読記録媒体。

【請求項14】 クライアントからサーバにリモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするスタブ検索ローディングプログラムを記録したコンピュータ可読記録媒体であって、

クライアントを構成するコンピュータを、スタブ名とクライアント識別子とを指定したスタブ要求をサーバに送出して、その応答として返されるスタブを受信するスタブ検索手段として機能させ、

サーバを構成するコンピュータを、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるク

クライアントの種類毎に生成するスタブ生成手段、クライアントからのスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ生成部に要求元のクライアントの実行環境に適したスタブを生成させ、生成されたスタブを要求元のクライアントに返すスタブ検索インターフェース手段として機能させるプログラムを記録したコンピュータ可読記録媒体。

【請求項15】 クライアントからサーバにリモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするスタブ検索ローディングプログラムを記録したコンピュータ可読記録媒体であって、

クライアントを構成するコンピュータを、スタブ名とクライアント識別子とを指定したスタブ要求をサーバに送出して、その応答として返されるスタブを受信するスタブ検索手段として機能させ、

サーバを構成するコンピュータを、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合の記憶手段、前記スケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に生成するスタブ生成手段、クライアントからのスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ集合の中から適切なスタブを検索して要求元のクライアントに返し、適切なスタブが前記スタブ集合に存在しなければ指定されたスタブ名とクライアント識別子に基づいて前記スタブ生成部に要求元のクライアントの実行環境に適したスタブを生成させ、生成されたスタブを要求元のクライアントに返すスタブ検索インターフェース手段として機能させるプログラムを記録したコンピュータ可読記録媒体。

【請求項16】 リモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをクライアントからのスタブ要求に応じて要求元のクライアントに提供するサーバ装置を構成するコンピュータを、

クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合の記憶手段、クライアントからのスタブ名とクライアント識別子とを指定したスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記ス

タブ集合の中から適切なスタブを検索して要求元のクライアントに返すスタブ検索インターフェース手段として機能させるプログラムを記録したコンピュータ可読記録媒体。

【請求項 1 7】 リモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをクライアントからのスタブ要求に応じて要求元のクライアントに提供するサーバ装置を構成するコンピュータを、

クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に生成するスタブ生成手段、クライアントからのスタブ名とクライアント識別子とを指定したスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ生成部に要求元のクライアントの実行環境に適したスタブを生成させ、生成されたスタブを要求元のクライアントに返すスタブ検索インターフェース手段として機能させるプログラムを記録したコンピュータ可読記録媒体。

【請求項 1 8】 リモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをクライアントからのスタブ要求に応じて要求元のクライアントに提供するサーバ装置を構成するコンピュータを、

クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合の記憶手段、前記スケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に生成するスタブ生成手段、クライアントからのスタブ名とクライアント識別子とを指定したスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ集合の中から適切なスタブを検索して要求元のクライアントに返し、適切なスタブが前記スタブ集合に存在しなければ指定されたスタブ名とクライアント識別子に基づいて前記スタブ生成部に要求元のクライアントの実行環境に適したスタブを生成させ、生成されたスタブを要求元のクライアントに返すスタブ検索インターフェース手段として機能させるプログラムを記録したコンピュータ可読記録媒体。

【請求項 1 9】 リモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするクライアント装置を構成するコ

ンピュータを、

クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合、及びクライアントからのスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ集合の中から適切なスタブを検索して要求元のクライアントに返すスタブ検索インターフェースとを備えたサーバ装置に対して、スタブ名とクライアント識別子とを指定したスタブ要求を送出し、その応答として返されるスタブを受信するスタブ検索手段として機能させるプログラムを記録したコンピュータ可読記録媒体。

【請求項 2 0】 リモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするクライアント装置を構成するコンピュータを、

クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に生成するスタブ生成部及び、クライアントからのスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ生成部に要求元のクライアントの実行環境に適したスタブを生成させ、生成されたスタブを要求元のクライアントに返すスタブ検索インターフェースとを備えたサーバ装置に対して、スタブ名とクライアント識別子とを指定したスタブ要求をサーバに送出し、その応答として返されるスタブを受信するスタブ検索手段として機能させるプログラムを記録したコンピュータ可読記録媒体。

【請求項 2 1】 リモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするクライアント装置を構成するコンピュータを、

クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合、前記スケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に生成するスタブ生成部及び、クライアントからのスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ集合の中から適切な

スタブを検索して要求元のクライアントに返し、適切なスタブが前記スタブ集合に存在しなければ指定されたスタブ名とクライアント識別子に基づいて前記スタブ生成部に要求元のクライアントの実行環境に適したスタブを生成させ、生成されたスタブを要求元のクライアントに返すスタブ検索インターフェースとを備えたサーバ装置に対して、スタブ名とクライアント識別子とを指定したスタブ要求をサーバに送出し、その応答として返されるスタブを受信するスタブ検索手段として機能させるプログラムを記録したコンピュータ可読記録媒体。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

本発明は、リモートメソッド呼び出し機構を備えたコンピュータシステムに関し、特にクライアント側で必要となるスタブをサーバ側からダウンロードして使用するコンピュータシステムにおけるスタブ検索ローディングシステム及びその方法に関する。

【0002】

【従来の技術】

リモートメソッド呼び出し (Remote Method Invocation; 以下、RMI と称す) は、別の Java 仮想マシン上にある Java オブジェクトのメソッドを、同じ Java 仮想マシン上にあるオブジェクトのメソッドと同じように呼び出せるようにした機構であり、Java を使って分散システムを構築する際に使われる。

【0003】

図10にRMIの仕組みの概要を示す。RMIは以下の手順で処理される。先ず、クライアントオブジェクトからサーバオブジェクトへの呼び出しは、クライアントに置かれたスタブに送られる((1))。スタブは、この呼び出しに対してマーシャリング処理などを行って、サーバのスケルトンに転送する((2))。スケルトンは、スタブから送られてきた呼び出しに対してアンマーシャリング処理などを行ってサーバオブジェクトへの呼び出しを生成し、サーバオブジェクトの呼び出しを行う((3))。ここで、マーシャリング処理とは、内部のデータ表現形式を

異なるシステム間で交換可能なデータ形式に変換する処理を言い、アンマーシャリング処理はその逆の処理を言う。

【 0 0 0 4 】

サーバオブジェクトは、呼び出されたメソッドを実行し、戻り値がある場合はそれをスケルトンに送る((4))。スケルトンは、送られた戻り値に対してマーシャリング処理などを施して、クライアントのスタブに送り返す((5))。スタブは、送られてきた戻り値に対しアンマーシャリング処理などを施し、呼び出し元のクライアントオブジェクトにメソッド呼び出しの戻り値として返す((6))。

【 0 0 0 5 】

このようにクライアントオブジェクトからサーバオブジェクトの呼び出しは、全てスタブを通じて行われる。つまり、スタブは、分散環境におけるクライアント側にあってサーバオブジェクトの代理人としての役割を持つ。一般には、1つのサーバオブジェクト毎に1つのスタブが必要となる。

【 0 0 0 6 】

スタブとスケルトンは、サーバアプリケーションの呼び出しインタフェースを記述したIDL (Interface Definition Language) ファイルをIDLコンパイラでコンパイルすることで、生成することができる。スタブとスケルトンを実際に利用するときは、クライアントアプリケーション、サーバアプリケーションをコンパイルするときに、スタブ、スケルトンをリンクして一緒にコンパイルする方式(以下これを静的方式と言う)が一般的である。しかし、静的方式では、リモート手続きを呼び出すのに必要な情報はプログラムの実行時でなくコンパイル時に決定しなければならない、プログラムにリンクされたスタブと、そのプログラムの実行時に呼び出されるリモート手続きの要件との間の不一致のためにエラーが発生し、効率が低下する恐れがあった。また、全てのスタブをクライアント側で記憶するための記憶容量が必要であった。

【 0 0 0 7 】

そこで、このような問題点を解消するために、クライアントがスタブをサーバから動的にダウンロードして使用する方式(以下これを動的方式と言う)が提案されている(例えば特開平10-83308号公報)。以下、従来の動的方式に

ついて、図面を参照して説明する。

【0008】

図11に動的方式を採用した従来のコンピュータシステムの一例を示す。この従来のコンピュータシステムは、クライアントコンピュータ101、102、サーバコンピュータ103及びネームサーバコンピュータ104が、ネットワーク105を通じて相互に接続されている。

【0009】

クライアントコンピュータ101には、Java実行環境 (Java Runtime Environment) 110が構築されており、このJava実行環境110の下で、クラス111が実行される。Java実行環境とは、Java仮想マシンを含めて、Javaプログラムを実行させるために必要となるソフトウェアである。また、クラス111は、Javaソースコードをコンパイルして作成されたクラスファイルであり、Java仮想マシンが実行するコードで記述されている。図では、スタブクラスを含め、クライアントコンピュータ101に存在するクラスの全てをクラス111で表現している。

【0010】

Java実行環境110には、制御部112と、スタブ検索部113を有するクラスローダ114と、インスタンス115とが含まれる。スタブ検索部113は、サーバコンピュータ103からスタブクラスをダウンロードする手段である。クラスローダ114は、クラス111及びスタブ検索部113でダウンロードしたスタブクラスをJava実行環境110にロードする手段である。インスタンス115は、クラスローダ114によってロードされたクラス111及びスタブクラスをインスタンス化して生成したオブジェクトである。図では、スタブクラスインスタンスを含め、Java実行環境110に存在するインスタンスの全てをインスタンス115で表現している。個々のインスタンス (オブジェクト) 115は、変数 (状態) とメソッド (振る舞い) とを持つ。周知のようにオブジェクト指向プログラミングにおいては、このような複数のオブジェクト (インスタンス) が互いにメッセージを送受信し合い、その相互作用によってプログラムが実行される。制御部112は、Java実行環境110の制御モジュールであ

り、J a v a 仮想マシン等を含んで構成される。制御部 1 1 2 は、クラスローダ 1 1 4 によってロードされたクラスをインスタンス化してインスタンス 1 1 5 を生成し、クライアント側のプログラムの実行を制御する。

#### 【0 0 1 1】

他のクライアントコンピュータ 1 0 2 も、クライアントコンピュータ 1 0 1 と同様な構成を有する。

#### 【0 0 1 2】

サーバコンピュータ 1 0 3 にも、J a v a 実行環境 1 2 0 が構築されており、この J a v a 実行環境 1 2 0 の下で、クラス 1 2 1 及びスケルトンクラス 1 2 2 が実行される。J a v a 実行環境 1 2 0 には、制御部 1 2 3 と、クラスローダ 1 2 4 と、インスタンス 1 2 5 とが含まれる。クラスローダ 1 2 4 は、クラス 1 2 1 及びスケルトンクラス 1 2 2 を J a v a 実行環境 1 2 0 にロードする手段である。インスタンス 1 2 5 は、クラスローダ 1 2 4 によってロードされたクラス 1 2 1 及びスケルトンクラス 1 2 2 をインスタンス化して生成したオブジェクトである。図では、スケルトンクラスインスタンスを含め、J a v a 実行環境 1 2 0 に存在するインスタンスの全てをインスタンス 1 2 5 で表現している。制御部 1 2 3 は、J a v a 実行環境 1 2 0 の制御モジュールであり、J a v a 仮想マシン等を含んで構成される。制御部 1 2 3 は、クラスローダ 1 2 4 によってロードされたクラス及びスケルトンクラスをインスタンス化してインスタンス 1 2 5 を生成し、サーバ側のプログラムの実行を制御する。

#### 【0 0 1 3】

サーバコンピュータ 1 0 3 には、更に、スタブクラス 1 2 6 とスタブ検索インターフェース 1 2 7 とが設けられている。スタブクラス 1 2 6 とスケルトンクラス 1 2 2 とは、1 対 1 に対応している。つまり、I D L ファイルをコンパイルして生成されたスケルトンクラス 1 2 2 とスタブクラス 1 2 6 のペアが、サーバコンピュータ 1 0 3 で保持管理されている。そして、これらのスタブクラス 1 2 6 は、スタブクラスの名前で検索可能に保持されている。スタブ検索インターフェース 1 2 7 は、クライアントコンピュータ 1 0 1 、 1 0 2 のスタブ検索部 1 1 3 からのスタブクラス名を指定したダウンロード要求時に、指定されたスタブクラ



ス名のスタブクラス126を検索し、それを要求元のスタブ検索部113に送り返す手段である。

#### 【0014】

ネームサーバコンピュータ104は、クライアントコンピュータ101、102が、必要とするスタブクラスをどのサーバコンピュータ103からダウンロードして良いか分からない、つまり、必要とするスタブクラスを保持するサーバコンピュータ103をクライアントコンピュータ101、102が知らない場合に、該当するサーバコンピュータ103をクライアントコンピュータ101、102に教える機能を持つコンピュータである。典型的には、スタブクラス名毎に、そのスタブクラスを保持するサーバコンピュータ103の識別子を保持するテーブルを保持管理しており、クライアントコンピュータ101、102からスタブクラス名を指定した問い合わせがあった際に、そのテーブルを参照して、該当するサーバコンピュータ103の識別子を返すように構成される。

#### 【0015】

次に従来のコンピュータシステムにおけるRMIの処理手順を、図12のフローチャートを参照して説明する。

#### 【0016】

クライアントコンピュータ101のJava実行環境110の制御部112は、或るクラス111のインスタンス115から、サーバコンピュータ103における或るクラス121の或るメソッドを呼び出すリモートメソッド呼び出しが行われると、その呼び出しに使うスタブクラス126のインスタンス115がJava実行環境110に存在するか否かを判別する(S101)。存在した場合には、図10で説明したような手順に従って、そのスタブクラス126のインスタンス115を使って当該リモートメソッド呼び出しを実現する(S102)。

#### 【0017】

今回の呼び出しに使うスタブクラス126のインスタンス115がJava実行環境110に存在しなかった場合、制御部112は、今回の呼び出しに使うスタブクラス126がクライアントコンピュータ101のクラス111中に存在するか否かを判別する(S103)。存在した場合、制御部112はクラスローダ

114によって、今回の呼び出しに使うスタブクラスのクラス111をJava実行環境110にロードし(S104)、それをインスタンス化して、今回の呼び出しに使うスタブクラス126のインスタンス115を生成する(S105)。そして、そのスタブクラス126のインスタンス115を使って当該リモートメソッド呼び出しを実現する(S102)。

## 【0018】

今回の呼び出しに使うスタブクラス126がクライアントコンピュータ101のクラス111中に存在しなかった場合、今回の呼び出しに使うスタブクラス126の存在場所を、クライアントコンピュータ101側で知っているか否かを判別する(S106)。今回の呼び出しに使うスタブクラス126の存在場所は、例えばその呼び出し中に含めることができ、またクラスローダ114で各スタブクラス126の所在を管理することもできるため、そのような場合にはクライアントコンピュータ101側で知っていることになる。他方、クライアントコンピュータ101側が今回の呼び出しに使うスタブクラス126の存在場所を知らなかった場合、制御部112は、スタブクラスのクラス名を指定してネームサーバコンピュータ104に問い合わせる(S107)。これに応答してネームサーバコンピュータ104は、指定されたクラス名のスタブクラスの存在場所を通知する(S108)。

## 【0019】

今回の呼び出しに使うスタブクラス126の存在場所が明らかになると、制御部112は、スタブ検索部113によって、前記存在場所で一意に定まるサーバコンピュータ103に対し、クラス名を指定してスタブクラスを要求する(S109)。サーバコンピュータ103のスタブ検索インターフェース127は、指定されたクラス名のスタブクラスをスタブクラス126から検索し、要求元のクライアントコンピュータ101に返す(S110、S111)。この返されてきたスタブクラスは、クラスローダ114によってJava実行環境110にロードされ(S104)、制御部112によってそのインスタンスが生成される(S105)。そして、そのスタブクラス126のインスタンス115を使って当該リモートメソッド呼び出しを実現する(S102)。なお、スタブ検索インター

フェース 1 2 7 による検索で該当するスタブクラスが見つからなかった場合、その旨がクライアントコンピュータ 1 0 1 に返される (S 1 1 2)。この場合、クライアントコンピュータ 1 0 1 側ではスタブクラスのダウンロード要求は失敗に終わり、例外処理によりリモートメソッド呼び出しはエラー終了する。

#### 【 0 0 2 0 】

##### 【発明が解決しようとする課題】

スタブクラスはマーシャリング処理等を行うスタブクラスインスタンスのクラスであり、実行されるクライアントコンピュータのマシンや OS や J a v a 実行環境などで特定される、スタブクラスの実行環境に依存した構成となる。このため、サーバコンピュータ 1 0 3 の J a v a 実行環境 1 2 0 の下で実行されるインスタンス 1 2 5 のメソッドを呼び出すのに必要なクライアント側のスタブクラスは、実行される J a v a 実行環境 1 1 0 の種類が違えば、そのクラス名は同じであっても、中身は異なる。

#### 【 0 0 2 1 】

図 1 1 で説明した従来のコンピュータシステムでは、クライアントコンピュータ 1 0 1 のスタブ検索部 1 1 3 はクラス名を指定してスタブクラスを要求し、サーバコンピュータ 1 0 3 のスタブ検索インターフェース 1 2 7 は指定されたクラス名を持つスタブクラスを返すようにしている。従って、クライアントコンピュータ 1 0 1 とクライアントコンピュータ 1 0 2 の J a v a 実行環境 1 1 0 の種類が同じであれば問題はないが、相違すれば、サーバコンピュータ 1 0 3 からダウンロードされるスタブクラスに適合する J a v a 実行環境を備えたクライアントコンピュータだけが正常に動作し、他のクライアントコンピュータは正常な動作を保証できない。これは、J a v a 実行環境の種類が異なるために同じスタブクラスを使用できない複数のクライアントコンピュータ 1 0 1、1 0 2 から同じサーバコンピュータ 1 0 3 に対して、動的方式によってスタブを獲得してリモートメソッド呼び出しを行うことができないことを意味する。また、J a v a 実行環境の種類が異なるために同じスタブクラスを使用できない複数のクライアントコンピュータからリモートメソッド呼び出しを行うことができるようにするには、複数のサーバコンピュータを用意しなければならないことを意味する。

## 【 0 0 2 2 】

本発明の目的は、1つのサーバコンピュータから、J a v a 実行環境の種類が異なる複数のクライアントコンピュータに対して、各クライアントコンピュータにおけるリモートメソッド呼び出しに使えるスタブクラスをダウンロードできるようにすることにある。

## 【 0 0 2 3 】

本発明の別の目的は、クライアントコンピュータから要求されたスタブクラスがサーバコンピュータに存在しない場合、適切なスタブクラスを動的に生成してダウンロードできるようにすることにある。

## 【 0 0 2 4 】

## 【課題を解決するための手段】

本発明の第1のスタブ検索ローディングシステムは、クライアントからサーバにリモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするスタブ検索ローディングシステムであって、

クライアント側に、スタブ名とクライアント識別子とを指定したスタブ要求をサーバに送出して、その応答として返されるスタブを受信するスタブ検索部を備え、

サーバ側に、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合と、クライアントからのスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ集合の中から適切なスタブを検索して要求元のクライアントに返すスタブ検索インターフェースとを備えている。

## 【 0 0 2 5 】

本発明の第2のスタブ検索ローディングシステムは、クライアントからサーバにリモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするスタブ検索ローディングシステムであって、

クライアント側に、スタブ名とクライアント識別子とを指定したスタブ要求をサーバに送出して、その応答として返されるスタブを受信するスタブ検索部を備え、

サーバ側に、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に生成するスタブ生成部と、クライアントからのスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ生成部に要求元のクライアントの実行環境に適したスタブを生成させ、生成されたスタブを要求元のクライアントに返すスタブ検索インターフェースとを備えている。

## 【 0 0 2 6 】

また本発明の第3のスタブ検索ローディングシステムは、クライアントからサーバにリモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするスタブ検索ローディングシステムであって、

クライアント側に、スタブ名とクライアント識別子とを指定したスタブ要求をサーバに送出して、その応答として返されるスタブを受信するスタブ検索部を備え、

サーバ側に、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合と、前記スケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に生成するスタブ生成部と、クライアントからのスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ集合の中から適切なスタブを検索して要求元のクライアントに返し、適切なスタブが前記スタブ集合に存在しなければ指定されたスタブ名とクライアント識別子に基づいて前記スタブ生成部に要求元のクライアントの実行環境に適したスタブを生成させ、生成されたスタブを要求元のクライアントに返すスタブ検索インターフェースとを備えている。

## 【 0 0 2 7 】

また本発明の第1のスタブ検索ローディング方法は、クライアントからサーバにリモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするスタブ検索ローディング方法であって、

(a) クライアントからサーバに、スタブ名とクライアント識別子とを指定したスタブ要求を送出するステップ、

(b) サーバにおいて、指定されたクライアント識別子に基づいて、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合の中から1つのスタブ集合を選択するステップ、

(c) サーバにおいて、指定されたスタブ名のスタブを前記選択したスタブ集合の中から選択するステップ、

(d) サーバからクライアントに、前記選択されたスタブを送出するステップ、

(e) クライアントにおいて、サーバから送付された前記スタブを受信するステップを含んでいる。

#### 【0028】

本発明の第2のスタブ検索ローディング方法は、クライアントからサーバにリモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするスタブ検索ローディング方法であって、

(a) クライアントからサーバに、スタブ名とクライアント識別子とを指定したスタブ要求を送出するステップ、

(b) サーバにおいて、指定されたスタブ名とクライアント識別子に基づいて、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを生成するステップ、

(c) サーバからクライアントに、前記生成したスタブを送出するステップ、

(d) クライアントにおいて、サーバから送付された前記スタブを受信するステップを含んでいる。

#### 【0029】

本発明の第3のスタブ検索ローディング方法は、クライアントからサーバにリモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするスタブ検索ローディング方法であって、

(a) クライアントからサーバに、スタブ名とクライアント識別子とを指定したスタブ要求を送出するステップ、

(b) サーバにおいて、指定されたクライアント識別子に基づいて、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使

うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合の中から1つのスタブ集合を選択し、且つ、該選択したスタブ集合の中から指定されたスタブ名のスタブを検索するステップ、

(c) サーバにおいて、ステップbの検索に失敗した場合に、指定されたスタブ名とクライアント識別子に基づいて、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを生成するステップ、

(d) サーバからクライアントに、ステップbの検索に成功した場合にはステップbで検索されたスタブを、ステップbの検索に失敗した場合にはステップcで生成されたスタブを送出するステップ、

(e) クライアントにおいて、サーバから送付された前記スタブを受信するステップを含んでいる。

#### 【0030】

ここで、クライアント識別子は、スタブが実行されるクライアントの実行環境をサーバ側で特定できるものであれば、どのようなものでも良い。例えば、クライアントコンピュータの機種名、型番、OSの種類、Java実行環境の種類、ネットワークアドレスや、これらに限定されずなにかしらのクライアントの情報、またはこれらの二つ以上を組み合わせたものでも良い。また、クライアントから明示的に送信される情報であっても良いし、ネットワークアドレスのようにスタブクラスの要求を送信することによって暗黙的に送信されるものであっても良い。

#### 【0031】

##### 【作用】

本発明の第1のスタブ検索ローディングシステム及びその方法にあっては、スタブを実行環境の異なるクライアントの種類毎にサーバ側に用意しておき、クライアントからサーバへのスタブ要求時にスタブ名に加えてそのクライアントの実行環境を特定し得るクライアント識別子を通知し、サーバ側では、指定されたスタブ名とクライアント識別子に基づいて適切なスタブを検索して要求元のクライアントに返すようにしたので、1つのサーバコンピュータから、実行環境の種類が異なる複数のクライアントコンピュータに対して適切なスタブをダウンロード

することが可能となる。

【 0 0 3 2 】

本発明の第2のスタブ検索ローディングシステム及びその方法にあつては、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に生成するスタブ生成部をサーバ側に用意しておき、クライアントからサーバへのスタブ要求時にスタブ名に加えてそのクライアントの実行環境を特定し得るクライアント識別子を通知し、サーバ側では、指定されたクライアント識別子のクライアントの実行環境に適したスタブを生成して要求元のクライアントに返すようにしたので、1つのサーバコンピュータから、実行環境の種類が異なる複数のクライアントコンピュータに対して適切なスタブをダウンロードすることが可能となる。

【 0 0 3 3 】

本発明の第3のスタブ検索ローディングシステム及びその方法にあつては、スタブを実行環境の異なるクライアントの種類毎にサーバ側に用意しておくと共に、スタブを実行環境の異なるクライアントの種類毎に生成するスタブ生成部をサーバ側に用意しておき、クライアントからサーバへのスタブ要求時にスタブ名に加えてそのクライアントの実行環境を特定し得るクライアント識別子を通知し、サーバ側では、指定されたスタブ名とクライアント識別子に基づいて適切なスタブを検索して要求元のクライアントに返すか、若し適切なスタブが存在しなければ、指定されたクライアント識別子のクライアントの実行環境に適したスタブを生成して要求元のクライアントに返すようにしたので、1つのサーバコンピュータから、実行環境の種類が異なる複数のクライアントコンピュータに対して適切なスタブをダウンロードすることが可能となる。

【 0 0 3 4 】

【発明の実施の形態】

次に本発明の実施の形態の例について図面を参照して詳細に説明する。

【 0 0 3 5 】

図1を参照すると、本発明の一実施の形態にかかるコンピュータシステムは、クライアントコンピュータ1～3、サーバコンピュータ4及びネームサーバコン



コンピュータ 5 が、ネットワーク 6 を通じて相互に接続されている。ネットワーク 6 は、LAN、WAN、公衆電話網、インターネットなど、コンピュータ間でデジタルデータを送受信できる任意のネットワークであり、その通信媒体も有線、無線或いはそれ以外のどのような媒体であっても構わない。

## 【 0 0 3 6 】

クライアントコンピュータ 1 には、Java 実行環境 (1) 1 0 が構築されており、この Java 実行環境 1 0 の下で、クラス 1 1 が実行される。クラス 1 1 は、Java ソースコードをコンパイルして作成されたクラスファイルであり、Java 仮想マシンが実行するコードで記述されている。図では、スタブクラスを含め、クライアントコンピュータ 1 に存在するクラスの全てをクラス 1 1 で表現している。

## 【 0 0 3 7 】

Java 実行環境 1 0 には、制御部 1 2 と、スタブ検索部 1 3 を有するクラスローダ 1 4 と、インスタンス 1 5 とが含まれる。スタブ検索部 1 3 は、サーバコンピュータ 4 からスタブクラスをダウンロードする手段である。クラスローダ 1 4 は、クラス 1 1 及びスタブ検索部 1 3 でダウンロードしたスタブクラスを Java 実行環境 1 0 にロードする手段である。インスタンス 1 5 は、クラスローダ 1 4 によってロードされたクラス 1 1 及びスタブクラスをインスタンス化して生成したオブジェクトである。図では、スタブクラスインスタンスを含め、Java 実行環境 1 0 に存在するインスタンスの全てをインスタンス 1 5 で表現している。制御部 1 2 は、Java 実行環境 1 0 の制御モジュールであり、Java 仮想マシン等を含んで構成される。制御部 1 2 は、クラスローダ 1 4 によってロードされたクラスをインスタンス化してインスタンス 1 5 を生成し、クライアント側のプログラムの実行を制御する。

## 【 0 0 3 8 】

他のクライアントコンピュータ 2, 3 も、クライアントコンピュータ 1 と同様な構成を有するが、コンピュータのマシン、OS などが各クライアントコンピュータ 1, 2, 3 で異なるため、構築されている Java 実行環境の種類が相違している。つまり、本例の場合、Java 実行環境 (1)、Java 実行環境 (2)、

J a v a 実行環境(3) の3種類の J a v a 実行環境があり、クライアントコンピュータ1には J a v a 実行環境(1) 1 0 が、クライアントコンピュータ2には J a v a 実行環境(2) 1 6 が、クライアントコンピュータ3には J a v a 実行環境(3) 1 7 が、それぞれ構築されている。

## 【 0 0 3 9 】

サーバコンピュータ4には、J a v a 実行環境(1) 2 0 が構築されており、この J a v a 実行環境2 0 の下で、クラス2 1 及びスケルトンクラス2 2 が実行される。J a v a 実行環境2 0 には、制御部2 3 と、クラスローダ2 4 と、インスタンス2 5 とが含まれる。クラスローダ2 4 は、クラス2 1 及びスケルトンクラス2 2 を J a v a 実行環境2 0 にロードする手段である。インスタンス2 5 は、クラスローダ2 4 によってロードされたクラス2 1 及びスケルトンクラス2 2 をインスタンス化して生成したオブジェクトである。図では、スケルトンクラスインスタンスを含め、J a v a 実行環境2 0 に存在するインスタンスを全てインスタンス2 5 で表現している。制御部2 3 は、J a v a 実行環境2 0 の制御モジュールであり、J a v a 仮想マシン等を含んで構成される。制御部2 3 は、クラスローダ2 4 によってロードされたクラス及びスケルトンクラスをインスタンス化してインスタンス2 5 を生成し、サーバ側のプログラムの実行を制御する。

## 【 0 0 4 0 】

サーバコンピュータ4には、更に、スタブクラス2 6 - 1 ~ 2 6 - 3 の記憶部2 9 と、クライアント識別部2 8 を有するスタブ検索インターフェース2 7 とが設けられている。スタブクラス2 6 - 1 は、J a v a 実行環境(1) を有するクライアントコンピュータ1においてリモートメソッド呼び出しを行う際に利用可能なスタブクラスを集めたものである。同様に、スタブクラス2 6 - 2、2 6 - 3 は、J a v a 実行環境(2)、(3) を有するクライアントコンピュータ2、3においてリモートメソッド呼び出しを行う際に利用可能なスタブクラスを集めたものである。各スタブクラス2 6 - 1 ~ 2 6 - 3 中の個々のスタブクラスは、スタブクラスの名前で検索可能に保持されている。また、同じリモートメソッド呼び出しに使うスタブクラス、つまり、サーバコンピュータ4における同じサーバオブジェクトのメソッドを呼び出すのに使うスタブクラスには、スタブクラス2 6 -

1～26-3で同じクラス名が付与されている。

【0041】

図2に、サーバコンピュータ4に保持するスタブクラス26-1～26-3を生成する構成例を示す。図2において、50-1～50-nは、サーバコンピュータ4が提供するサービスのインターフェースをJavaのインターフェースクラス（メソッドの名前と型の情報だけを持ち、メソッドの実装コードを持たないクラス）で定義したもので、サーバコンピュータ4が提供するサービス毎に存在する。

【0042】

RMIコンパイラ51-1は、JavaのインターフェースクラスからJava実行環境(1)用のスケルトンクラス及びスタブクラスを生成するコンパイラであり、このRMIコンパイラ51-1にインターフェースクラス50-1～50-nのそれぞれをコンパイルさせると、図1のサーバコンピュータ4に保持すべきスケルトンクラス22及びスタブクラス26-1が生成される。また、RMIコンパイラ51-2は、JavaのインターフェースクラスからJava実行環境(2)用のスケルトンクラス及びスタブクラスを生成するコンパイラであり、このRMIコンパイラ51-2にインターフェースクラス50-1～50-nのそれぞれをコンパイルさせると、図1のサーバコンピュータ4に保持すべきスタブクラス26-2が生成される。同時に生成されるスケルトンクラス52は使用しない。更に、RMIコンパイラ51-3は、JavaのインターフェースクラスからJava実行環境(3)用のスケルトンクラス及びスタブクラスを生成するコンパイラであり、このRMIコンパイラ51-3にインターフェースクラス50-1～50-nのそれぞれをコンパイルさせると、図1のサーバコンピュータ4に保持すべきスタブクラス26-3が生成される。同時に生成されるスケルトンクラス53は使用しない。RMIコンパイラ51-1～51-3としては、例えばrmicを使用することができる。

【0043】

図2の構成例では、1つのRMIコンパイラで1つのJava実行環境用のスタブクラスを生成したが、RMIコンパイラの種類によっては、作成するスタブ

クラスの種類をコンパイラオプションで指定できるものがあり、そのような RMI コンパイラを使えば、1つの RMI コンパイラで複数の J a v a 実行環境用のスタブクラスを生成することができる。例えば r m i c では、コンパイラオプションの指定により、バージョン 1.1 の J RMP スタブプロトコルバージョンのスタブとスケルトン、バージョン 1.2 の J RMP スタブプロトコルバージョンのスタブとスケルトン、バージョン 1.1、1.2 の両方の J RMP スタブプロトコルバージョンと互換性のあるスタブとスケルトンを作成することができる。図 3 に、そのような機能を持つ RMI コンパイラ 5 4 を使って複数の J a v a 実行環境用のスタブクラスを生成する構成例を示す。

## 【 0 0 4 4 】

図 3 において、RMI コンパイラ 5 4 に、J a v a 実行環境 (1) 用のオプション指定（例えばバージョン 1.1、1.2 の両方の J RMP スタブプロトコルバージョンと互換性のあるスタブとスケルトンを作成する指定）を与えて、インターフェースクラス 5 0 - 1 ~ 5 0 - n のそれぞれをコンパイルさせると、図 1 のサーバコンピュータ 4 に保持すべきスケルトンクラス 2 2 及びスタブクラス 2 6 - 1 が生成される。また、RMI コンパイラ 5 4 に、J a v a 実行環境 (2) 用のオプション指定（例えばバージョン 1.1 の J RMP スタブプロトコルバージョンのスタブとスケルトンを作成する指定）を与えて、インターフェースクラス 5 0 - 1 ~ 5 0 - n のそれぞれをコンパイルさせると、図 1 のサーバコンピュータ 4 に保持すべきスタブクラス 2 6 - 2 が生成される。同時に生成されるスケルトンクラス 5 2 は使用しない。更に、RMI コンパイラ 5 4 に、J a v a 実行環境 (3) 用のオプション指定（例えばバージョン 1.2 の J RMP スタブプロトコルバージョンのスタブとスケルトンを作成する指定）を与えて、インターフェースクラス 5 0 - 1 ~ 5 0 - n のそれぞれをコンパイルさせると、図 1 のサーバコンピュータ 4 に保持すべきスタブクラス 2 6 - 3 が生成される。同時に生成されるスケルトンクラス 5 3 は使用しない。

## 【 0 0 4 5 】

再び図 1 を参照すると、クライアント識別部 2 8 は、クライアントコンピュータ 1 ~ 3 からのスタブクラス名を指定したスタブ要求時に、同時に送られてくる

クライアント識別子を識別し、スタブクラス 26-1～26-3 の内、どのスタブクラスの集合を検索対象とすべきかを決定する部分である。クライアント識別部 28 は、例えば、クライアント識別子とスタブクラスの集合名（スタブクラス 26-1～26-3 の名前）との対応を保持するテーブル（図示せず）を保持しており、このテーブルをクライアント識別子で検索して該当するスタブクラスの集合を特定する。スタブ検索インターフェース 27 は、クライアント識別部 28 で決定された何れか 1 つのスタブクラス 26-1～26-3 から、スタブ要求で指定されたスタブクラス名を持つスタブクラスを検索し、それを要求元のクライアントコンピュータ 1～3 に送り返す手段である。

## 【0046】

ネームサーバコンピュータ 5 は、クライアントコンピュータ 1～3 からのスタブクラス名を指定した問い合わせに応答して、例えばスタブクラス名毎にそのスタブクラスの所在場所情報を保持管理するテーブルを参照して、そのスタブクラスの所在場所を見つけ、それを問い合わせ元のクライアントコンピュータ 1～3 に返す機能を持つコンピュータである。

## 【0047】

次に、図 1 に示したコンピュータシステムにおける RMI の処理手順を、本発明の特徴部分であるスタブクラスのダウンロード動作を中心に図 4 のフローチャートを参照して説明する。

## 【0048】

クライアントコンピュータ 1 の J a v a 実行環境 10 の制御部 12 は、例えば或るクラス 11 のインスタンス 15 から、サーバコンピュータ 4 における或るクラス 21 の或るメソッドを呼び出すリモートメソッド呼び出しが行われると、その呼び出しに使うスタブクラスのインスタンスが J a v a 実行環境 10 のインスタンス 15 中に存在するか否かを判別する（S1）。存在した場合には、図 10 で説明したような手順に従って、そのスタブクラスのインスタンス 115 を使って当該リモートメソッド呼び出しを実現する（S2）。

## 【0049】

今回の呼び出しに使うスタブクラスのインスタンス 15 が J a v a 実行環境 1

0に存在しなかった場合、制御部12は、今回の呼び出しに使うスタブクラスがクライアントコンピュータ1のクラス11中に存在するか否かを判別する(S3)。存在した場合、制御部12はクラスローダ14によって、今回の呼び出しに使うスタブクラスのクラス11をJ a v a 実行環境10にロードし(S4)、それをインスタンス化して、今回の呼び出しに使うスタブクラスのインスタンス15を生成する(S5)。そして、そのスタブクラスのインスタンス15を使って当該リモートメソッド呼び出しを実現する(S2)。

## 【0050】

今回の呼び出しに使うスタブクラスがクライアントコンピュータ1のクラス11中に存在しなかった場合、今回の呼び出しに使うスタブクラスの存在場所を、クライアントコンピュータ1側で知っているか否かを従来例で説明したと同様に判別する(S6)。知っている場合にはステップS9へ進み、知らなかった場合は、スタブクラスのクラス名を指定してネームサーバコンピュータ5に問い合わせる(S7)。これに応答してネームサーバコンピュータ5は、指定されたクラス名のスタブクラスの存在場所を通知する(S8)。

## 【0051】

今回の呼び出しに使うスタブクラスの存在場所が明らかになると、制御部12は、スタブ検索部13によって、前記存在場所で一意に定まるサーバコンピュータ4に対し、クラス名及びクライアント識別子を指定してスタブクラスを要求する(S9)。ここで、クライアント識別子は、クライアントコンピュータ1の機種または型番またはOSの種類またはJ a v a 実行環境(1)の種類または制御部12の種類またはネットワークアドレスまたはこれらに限定されずなにかしらのクライアントの情報、またはこれらの二つ以上を組み合わせたもので、スタブ検索部13によって明示的に送信される情報であっても、ネットワークアドレスのようにスタブクラスの要求を送信することによって暗黙的に送信されてサーバコンピュータ4のクライアント識別部28がその情報によってクライアントコンピュータを識別することができる情報であってもよい。要するに、クライアント識別子は、ダウンロードしたスタブが実行される実行環境を直接的または間接的に特定できるものであれば良い。

## 【 0 0 5 2 】

サーバコンピュータ4のスタブ検索インターフェース27は、クライアント識別部28によって、受信したクライアント識別子に基づいてスタブクラス26-1~26-3の内、どのスタブクラスの集合を検索対象とすべきかを決定し、この決定した何れか1つのスタブクラス26-1~26-3から、スタブ要求で指定されたスタブクラス名を持つスタブクラスを検索する(S10)。今の場合には、J a v a実行環境(1)用のスタブクラス26-1からスタブクラスが検索される。そして、該当するスタブクラスが存在した場合には、それを要求元のクライアントコンピュータ1に送り返す(S11)。また、存在しなかった場合には、クライアントコンピュータ1に対して該当するスタブクラスを保持していない旨を通知する(S12)。この場合、スタブクラスのダウンロード要求は失敗に終わり、クライアントコンピュータ1では例外処理が実施され、例えばリモートメソッド呼び出しがエラー終了する。

## 【 0 0 5 3 】

クライアントコンピュータ1のスタブ検索部13は、要求先のサーバコンピュータ4からスタブクラスが送られてくると、これをクラスローダ14に渡し、クラスローダ14はJ a v a実行環境10にロードする(S4)。そして、制御部12はこのロードされたスタブクラスをインスタンス化してインスタンス15を生成し(S5)、インスタンス15を使って当該リモートメソッド呼び出しを実現する(S2)。

## 【 0 0 5 4 】

以上は、クライアントコンピュータ1を例にして動作を説明したが、他のクライアントコンピュータ2、3におけるRMIの処理も同様に行われる。但し、クライアントコンピュータ2、3におけるサーバコンピュータ4へのスタブクラスの要求時には、クライアントコンピュータ2、3のクライアント識別子が送られるので、サーバコンピュータ4のスタブ検索インターフェース27は、クライアントコンピュータ2からのスタブクラス要求時にはJ a v a実行環境(2)用のスタブクラス26-2を検索し、クライアントコンピュータ3からのスタブクラス要求時にはJ a v a実行環境(3)用のスタブクラス26-3を検索する。

## 【0055】

このようにして本実施の形態にかかるコンピュータシステムは、1つのサーバコンピュータ4から、Java実行環境の種類が異なる複数のクライアントコンピュータ1～3に対して、各クライアントコンピュータ1～3におけるリモートメソッド呼び出しに使えるスタブクラスをダウンロードし、同じサーバオブジェクトのメソッドに対するリモートメソッド呼び出しを可能にしている。このため、構成が様々に異なるクライアントコンピュータ1～3に対して同じ入口のサービスを一つだけサーバコンピュータに用意しておくだけで済ませることができる。

## 【0056】

次に、具体的な実施例を用いて本実施の形態の動作を補足説明する。クライアントコンピュータ1上のクライアントFooがサーバにリモートメソッド呼び出しを行うにあたってBankService インターフェースを実装したクラスであるBankServiceImpl クラスのスタブクラスが必要になった時、クライアントコンピュータ1はネームサーバコンピュータ5にBankServiceImpl のスタブクラスがどこにあるかを問い合わせると、`http://japan/BankServiceImpl Stub.class`と返って来た。

## 【0057】

そこで、クライアントコンピュータ1は、japan サーバに対して、HTTPプロトコルを使用して、HTTPヘッダにクライアント識別子として "User-Agent: Foo" という文字列を付加して `http://japan/BankServiceImpl Stub.class`を要求する。

## 【0058】

サーバコンピュータ4は、Foo という種類のクライアント識別子にマッチするものとして `http://japan/client/Foo/BankServerImpl Stub.class`を取り出してクライアントコンピュータ1に返す。

## 【0059】

一方、クライアントコンピュータ2上のクライアントBarがサーバにリモートメソッド呼び出しを行う際は、ネームサーバコンピュータに問い合わせ `http://japan/BankServiceImpl Stub.class`とかえって来る所までは同様である。次に、japan サーバに対して、HTTP プロトコルを使用して japanサーバに `http://`



japan/BankServiceImpl Stub.classを要求する際、HTTPヘッダにクライアント識別子として "User-Agent: Bar"という文字列を付加する。サーバコンピュータ4は、今度は Barという種類のクライアント識別子にマッチするものとして http://japan/client/Bar/BankServerImpl Stub.classを取り出してクライアントコンピュータ2に返す。

#### 【0060】

クライアントコンピュータ1及び2にダウンロードされインスタンス化されたスタブクラスは、いずれも japanサーバの BankServiceImplのスケルトンと通信し、リモートメソッド呼び出しを実現する。

#### 【0061】

#### 【発明の他の実施の形態】

図5は本発明の別の実施の形態にかかるコンピュータシステムのブロック図であり、図1と同一符号は同一部分を示し、30はスタブ生成部である。本実施の形態が図1に示した実施の形態と相違するところは、サーバコンピュータ4が、実行環境の異なるクライアントの種類毎のスタブを生成するスタブ生成部30を備え、スタブクラスを要求したクライアントコンピュータ1～3に返却すべき適切なスタブクラスが記憶部29内のスタブクラス26-1～26-3中に存在しなかった場合、スタブ検索インターフェース27がスタブクラス名とクライアント識別子とを指定したスタブ生成要求をスタブ生成部30に出し、スタブ生成部30が該当するスタブを動的に生成し、スタブ検索インターフェース27がその生成されたスタブを要求元のクライアントコンピュータ1～3に返すようにした点にある。また、新たに生成されたスタブクラスを後のスタブ要求に備えて記憶部29に保存するようにした点にある。

#### 【0062】

図6にスタブ生成部30の構成例を示す。図6において、60-1～60-mは、サーバコンピュータ4が提供するサービスのインターフェースをJavaのインターフェースクラス（メソッドの名前と型の情報だけを持ち、メソッドの実装コードを持たないクラス）で定義したもので、サーバコンピュータ4が提供するサービス毎に存在する。61-1～61-nは、図2及び図3で説明したもの

と同様な RMI コンパイラであり、それぞれ、J a v a のインターフェースクラスからスケルトンクラス及びスタブクラスを生成する。62 は、スタブ検索インターフェース 27 から、スタブクラス名とクライアント識別子とを指定したスタブ生成要求を受け取って各部を制御する制御部である。

#### 【0063】

制御部 62 は、各インターフェースクラス 60-1 ~ 60-m がどのスタブクラス名に対応するものであるかを内部のテーブル（図示せず）によって管理すると共に、クライアント識別子を識別して、そのクライアント識別子のクライアントの実行環境で実行するスタブを生成するには複数存在する RMI コンパイラ 61-1 ~ 61-n のうち、どれを使用すべきであるか、またどのようなコンパイラオプションを指定すべきであるかを、例えばクライアント識別子と RMI コンパイラ 61-1 ~ 61-n との対応表（図示せず）によって管理している。そして制御部 62 は、スタブ検索インターフェース 27 からのスタブ生成要求で指定されたスタブクラス名によって選択したインターフェースクラス 60-1 ~ 60-m の 1 つを、同じくスタブ生成要求で指定されたクライアント識別子によって選択した RMI コンパイラ 61-1 ~ 61-n の 1 つに入力して（必要ならばコンパイラオプションも指定する）、コンパイルを行わせ、その RMI コンパイラで生成されたスケルトンクラスとスタブクラスのうち、スタブクラスを要求元のスタブ検索インターフェース 27 に返却する処理を行う。なお、制御部 62 は、該当するインターフェースクラスが存在しないか、該当する RMI コンパイラが存在しないため、該当するスタブを生成することができない場合には、その旨をスタブ検索インターフェース 27 に通知する。

#### 【0064】

次に、本実施の形態にかかるコンピュータシステムにおける RMI の処理手順を、図 1 の実施の形態との相違点を中心に説明する。

#### 【0065】

図 7 は本実施の形態にかかるコンピュータシステムにおける RMI の処理手順の一例を示すフローチャートであり、図 4 と同一符号は同一ステップを示す。図 4 と比較すると分かるように、本実施の形態では、スタブ検索インターフェース

27によるスタブクラスの検索で該当するスタブクラスがスタブクラス26-1～26-3中に存在しないと判断された場合（S10でNO）、スタブ検索インターフェース27からスタブクラス名とクライアント識別子とを通知されてスタブ生成部30が起動され、スタブ生成部30の制御部62が、インターフェースクラス60-1～60-m中にスタブクラス名に対応するインターフェースクラスが存在するか否か、RMIコンパイラ61-1～61-n中にクライアント識別子に対応するRMIコンパイラが存在するか否かを判定することによって、要求されたスタブクラスを生成できるか否かを調べる（S21）。そして、生成可能な場合、スタブ生成部30の制御部62の制御の下に該当するスタブクラスの動的な生成が行われ、生成されたスタブクラスがスタブ検索インターフェース27に戻される（S22）。スタブ検索インターフェース27はそれを要求元のクライアントコンピュータ1～3に返すと共に記憶部29に保存する（S11）。他方、スタブ生成部30で該当するスタブクラスが生成できない場合、その旨がスタブ生成部30からスタブ検索インターフェース27に返され、スタブ検索インターフェース27は要求元のクライアントコンピュータ1～3に該当するスタブクラスが存在しない旨を通知する（S12）。

## 【0066】

このようにして本実施の形態にかかるコンピュータシステムは、1つのサーバコンピュータ4から、Java実行環境の種類が異なる複数のクライアントコンピュータ1～3に対して、各クライアントコンピュータ1～3におけるリモートメソッド呼び出しに使えるスタブクラスを必要に応じて動的に生成してダウンロードし、同じサーバオブジェクトのメソッドに対するリモートメソッド呼び出しを可能にしている。このため、構成が様々に異なるクライアントコンピュータ1～3に対して同じ入口のサービスを一つだけサーバコンピュータに用意しておくだけで済ませることができ、また、予め全ての構成に対応するスタブクラスをサーバコンピュータ4に用意しておく必要がなくなる。

## 【0067】

なお、本実施の形態では、記憶部29にスタブクラスが1つも記憶されない状態でサーバコンピュータ4を立ち上げるようにしても良いし、サーバコンピュ-

タ 4 の立ち上げ時点で図 1 の実施の形態と同様に記憶部 2 9 に幾つかのスタブクラスを予め記憶させておくようにしても良い。また、スタブ生成部 3 0 で生成されたスタブクラスを記憶部 2 9 に保存するようにしたが、保存しないようにしても良い。

#### 【 0 0 6 8 】

##### 【発明の他の実施の形態】

図 8 は本発明の更に別の実施の形態にかかるコンピュータシステムのブロック図であり、図 5 と同一符号は同一部分を示す。本実施の形態が図 5 に示した実施の形態と相違するところは、サーバコンピュータ 4 が、スタブクラスの集合を一切保持せずに、クライアントコンピュータ 1 ～ 3 からダウンロード要求されたスタブクラスの全てを動的に生成するようにした点にある。そのため、図 5 のスタブクラス 2 6 - 1 ～ 2 6 - 3 の記憶部 2 9 は省略されている。

#### 【 0 0 6 9 】

スタブ検索インターフェース 2 7 は、クライアントコンピュータ 1 ～ 3 からスタブクラス名及びクライアント識別子を指定したスタブ要求を受けた場合、スタブ生成部 3 0 に対してスタブクラス名とクライアント識別子を通知してスタブクラスの生成を指示する。スタブ生成部 3 0 は図 5 の実施の形態で説明したものと同様の方法でスタブクラスを生成してスタブ検索インターフェース 2 7 に返し、スタブ検索インターフェース 2 7 はそれを要求元のクライアントコンピュータ 1 ～ 3 に送る。スタブ生成部 3 0 が該当するスタブクラスを生成できない場合、スタブ検索インターフェース 2 7 はスタブクラスが見つからない旨を要求元のクライアントコンピュータ 1 ～ 3 に通知する。

#### 【 0 0 7 0 】

図 9 は本実施の形態にかかるコンピュータシステムにおける R M I の処理手順の一例を示すフローチャートであり、図 7 と同一符号は同一ステップを示す。図 7 と比較すると分かるように、本実施の形態では、クライアントコンピュータからスタブ要求を受けた場合、スタブ検索インターフェース 2 7 からスタブ生成部 3 0 が直ちに起動され、スタブ生成部 3 0 は、スタブクラスを生成できるか否かを調べる ( S 2 1 ) 。生成可能な場合、スタブ生成部 3 0 は該当するスタブクラ

スを生成してスタブ検索インターフェース 2 7 に渡し (S 2 2)、スタブ検索インターフェース 2 7 はそれを要求元のクライアントコンピュータ 1 ~ 3 に返す (S 1 1)。他方、該当するスタブクラスが生成できない場合、その旨がスタブ生成部 3 0 からスタブ検索インターフェース 2 7 に通知され、スタブ検索インターフェース 2 7 は要求元のクライアントコンピュータ 1 ~ 3 に該当するスタブクラスが存在しない旨を通知する (S 1 2)。

#### 【0 0 7 1】

このようにして本実施の形態にかかるコンピュータシステムは、1 つのサーバコンピュータ 4 から、J a v a 実行環境の種類が異なる複数のクライアントコンピュータ 1 ~ 3 に対して、各クライアントコンピュータ 1 ~ 3 におけるリモートメソッド呼び出しに使えるスタブクラスを動的に生成してダウンロードし、同じサーバオブジェクトのメソッドに対するリモートメソッド呼び出しを可能にしている。このため、構成が様々に異なるクライアントコンピュータ 1 ~ 3 に対して同じ入口のサービスを一つだけサーバコンピュータに用意しておくだけで済ませることができ、また、スタブクラスをサーバコンピュータ 4 に保存しておく必要がなくなる。

#### 【0 0 7 2】

##### 【発明の効果】

以上説明したように本発明によれば、1 つのサーバコンピュータから、実行環境の種類が異なる複数のクライアントコンピュータに対して適切なスタブをダウンロードすることが可能となる。また、スタブ生成部によってスタブを動的に生成する構成にあっては、予め全ての構成に対応するスタブをサーバ側に用意しておく必要がなくなる。

##### 【図面の簡単な説明】

#### 【図 1】

本発明の一実施の形態にかかるコンピュータシステムのブロック図である。

#### 【図 2】

サーバコンピュータに保持するスタブクラスを生成する構成例を示すブロック図である。

【図 3】

サーバコンピュータに保持するスタブクラスを生成する別の構成例を示すブロック図である。

【図 4】

本発明の一実施の形態にかかるコンピュータシステムにおける RMI の処理手順を示すフローチャートである。

【図 5】

本発明の別の実施の形態にかかるコンピュータシステムのブロック図である。

【図 6】

スタブ生成部の構成例を示すブロック図である。

【図 7】

本発明の別の実施の形態にかかるコンピュータシステムにおける RMI の処理手順を示すフローチャートである。

【図 8】

本発明の更に別の実施の形態にかかるコンピュータシステムのブロック図である。

【図 9】

本発明の更に別の実施の形態にかかるコンピュータシステムにおける RMI の処理手順を示すフローチャートである。

【図 10】

RMI の仕組みの概要を示す図である。

【図 11】

従来のコンピュータシステムのブロック図である。

【図 12】

従来のコンピュータシステムにおける RMI の処理手順を示すフローチャートである。

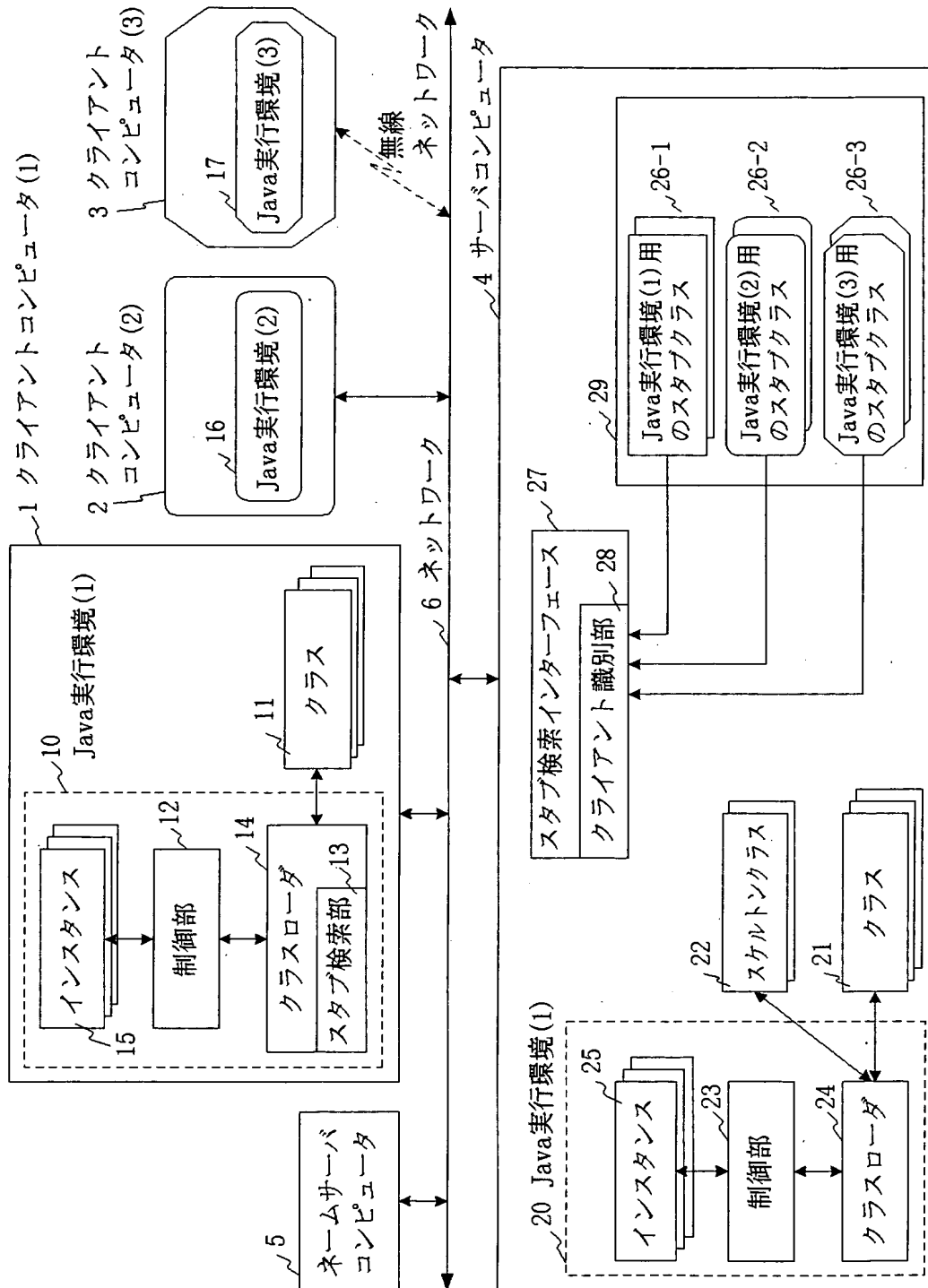
【符号の説明】

- 1、2、3…クライアントコンピュータ
- 4…サーバコンピュータ

5 … ネームサーバコンピュータ  
6 … ネットワーク  
1 0、1 6、1 7、2 0 … J a v a 実行環境  
1 1、2 1 … クラス  
1 2、2 3 … 制御部  
1 3 … スタブ検索部  
1 4、2 4 … クラスローダ  
1 5、2 5 … インスタンス  
2 2 … スケルトンクラス  
2 6 - 1 ~ 2 6 - 3 … スタブクラス  
2 7 … スタブ検索インターフェース  
2 8 … クライアント識別部  
2 9 … 記憶部  
3 0 … スタブ生成部

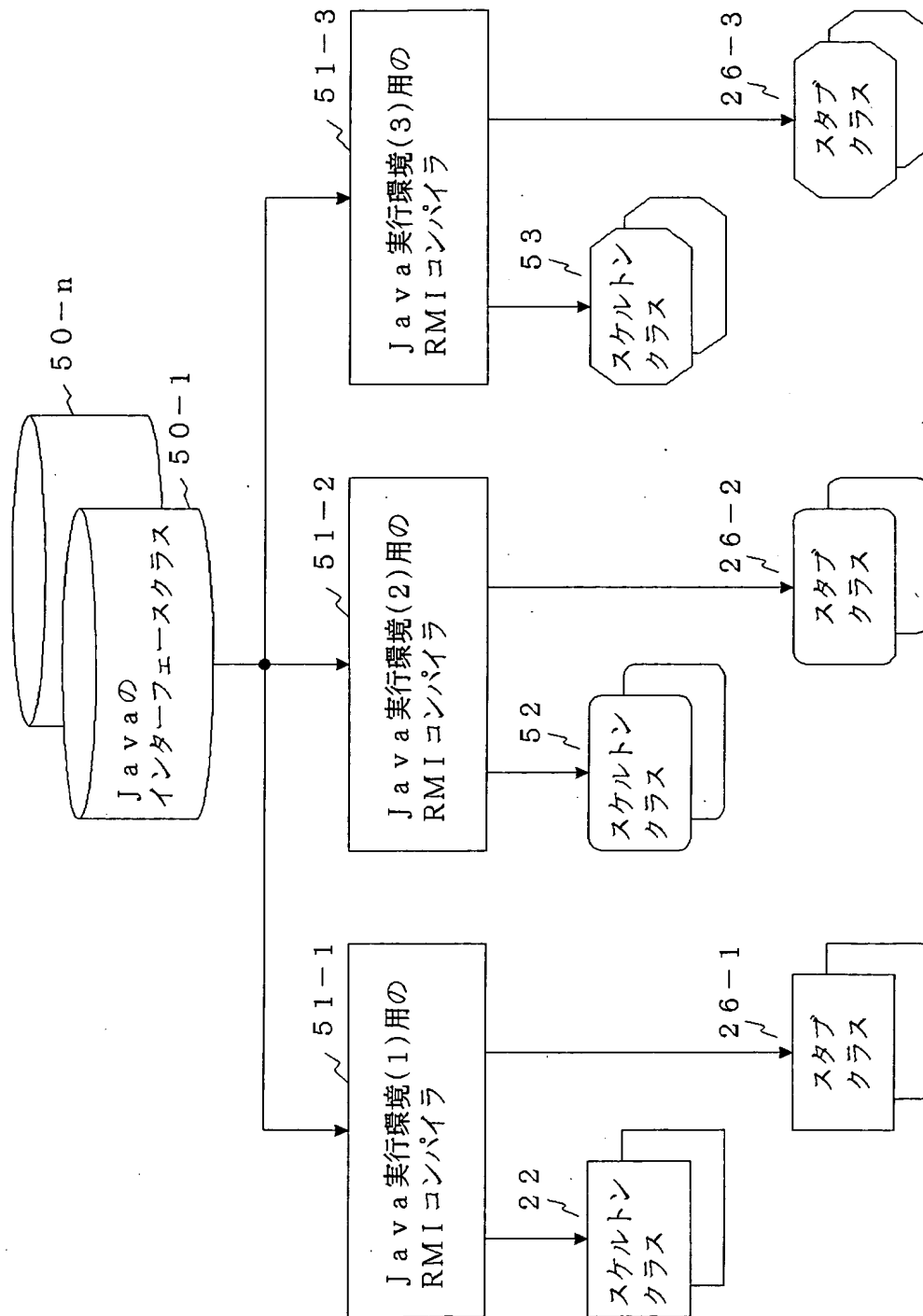
【書類名】 図面

【図 1】

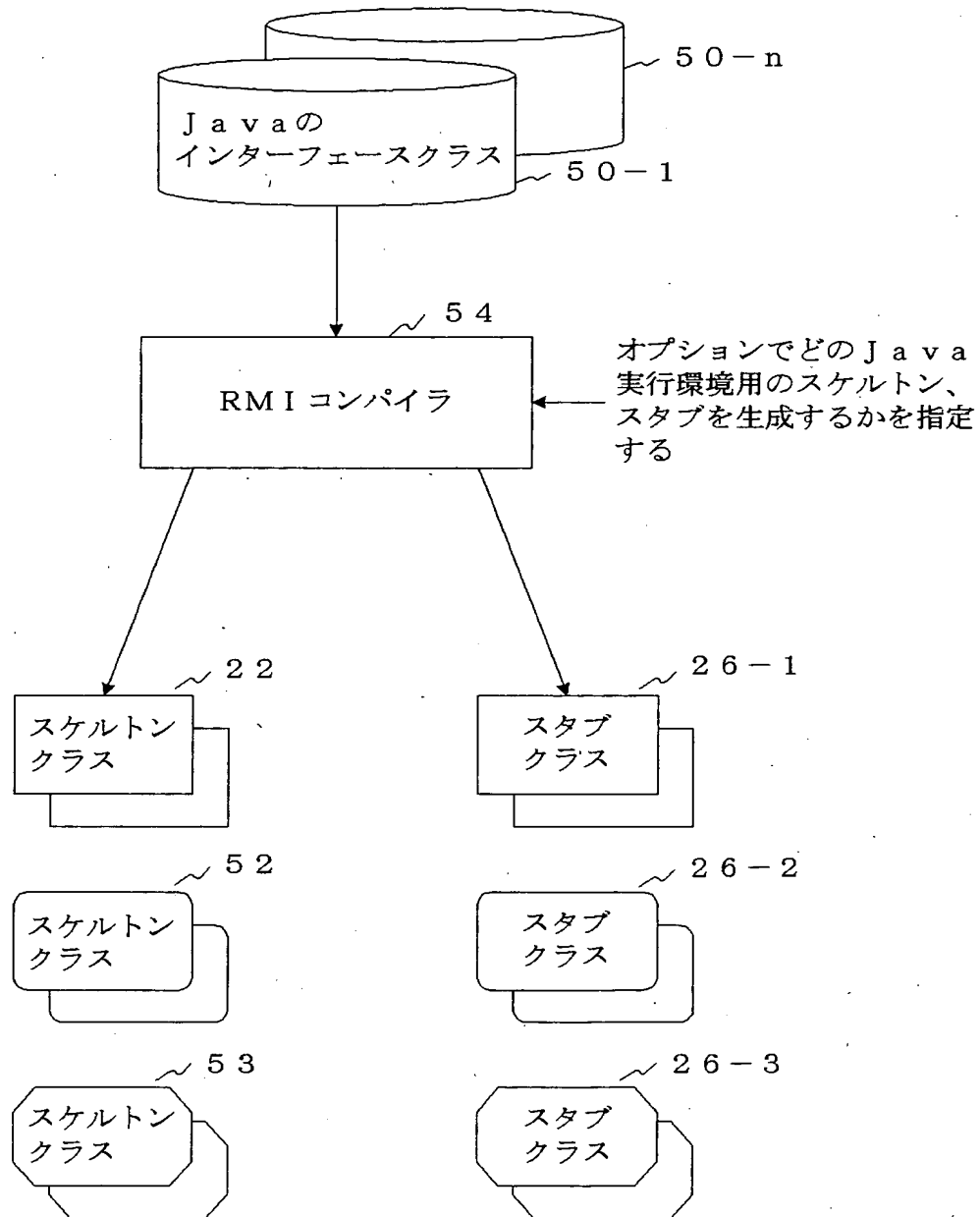




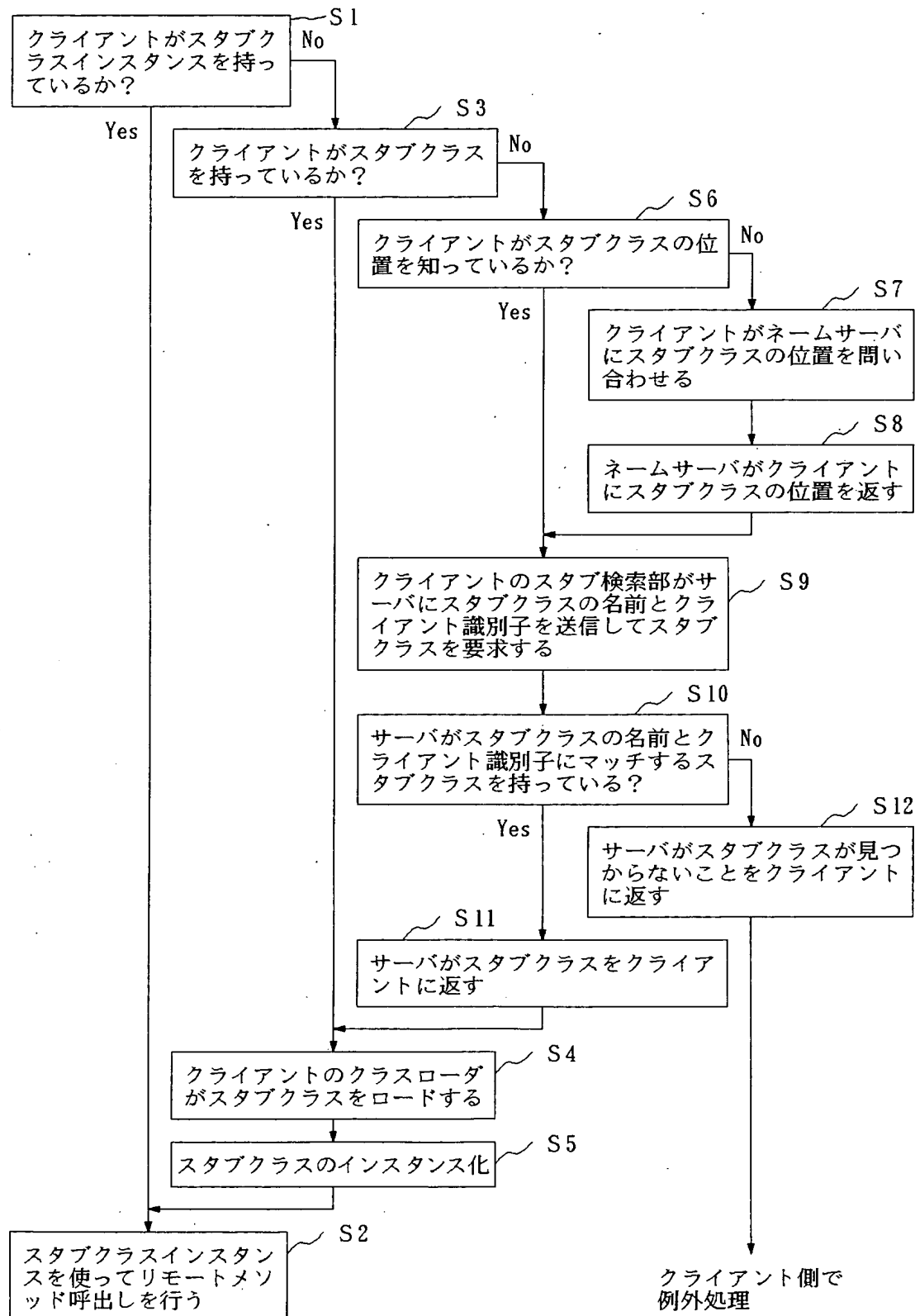
【図 2】



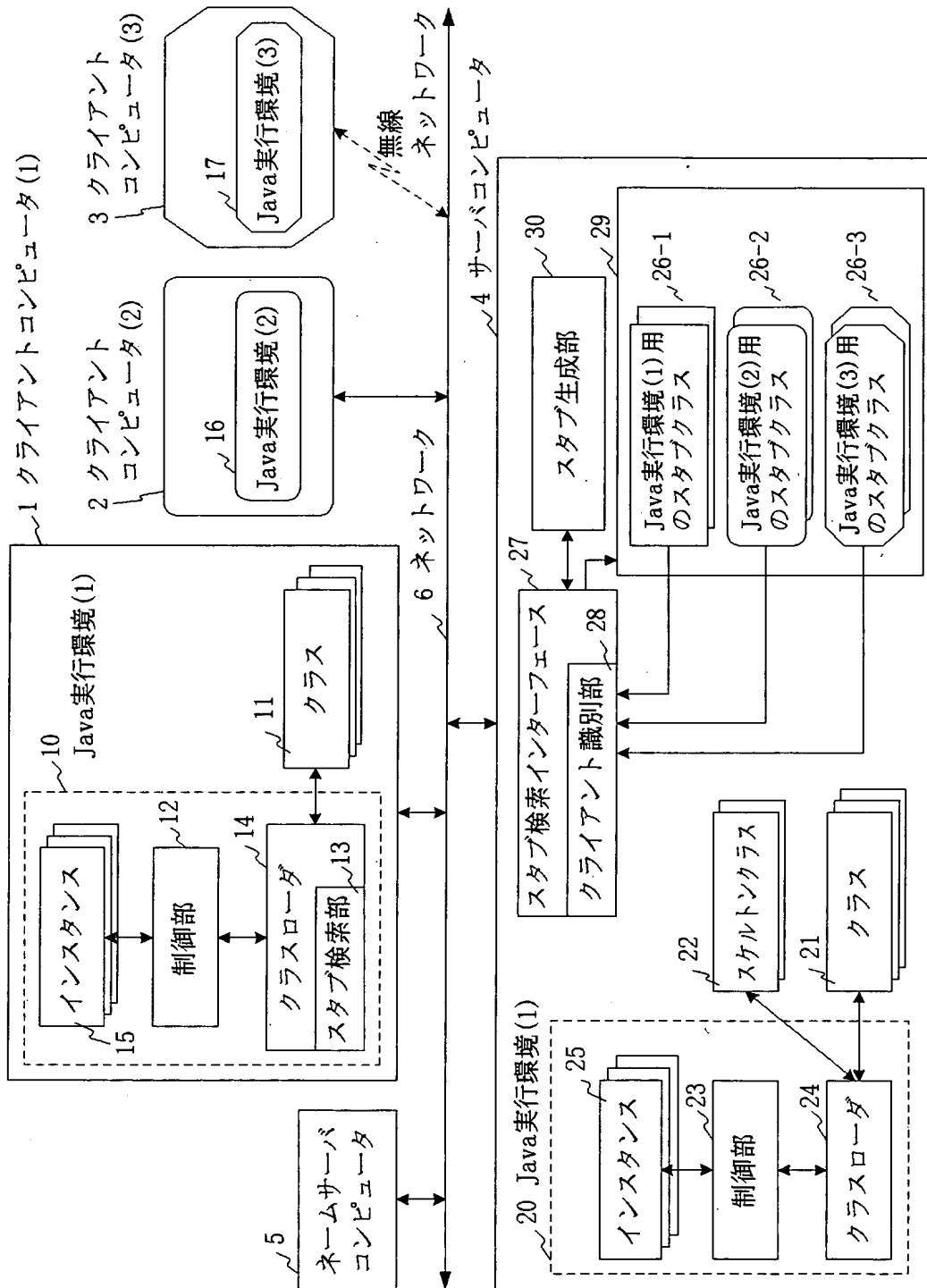
【図 3】



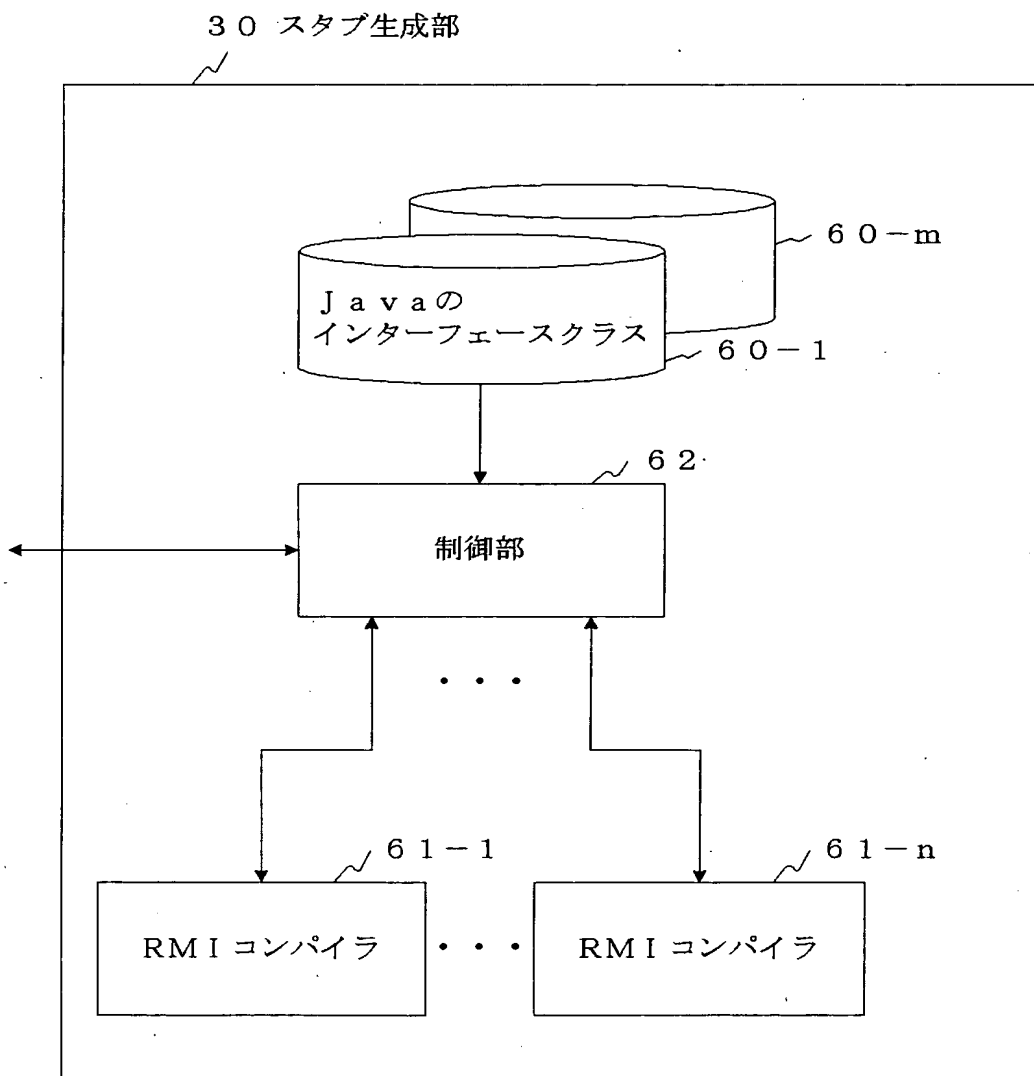
【図 4】



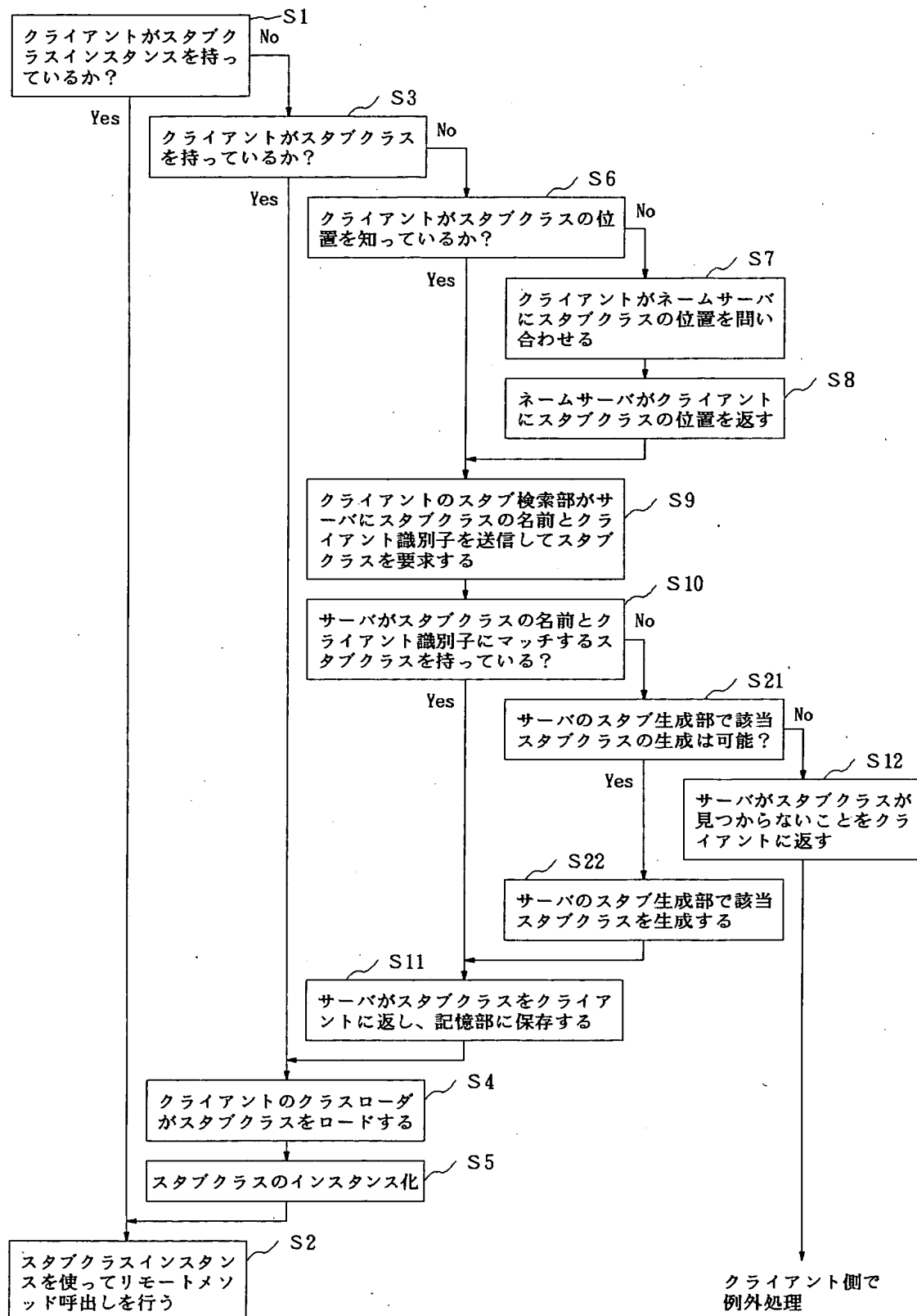
【図 5】



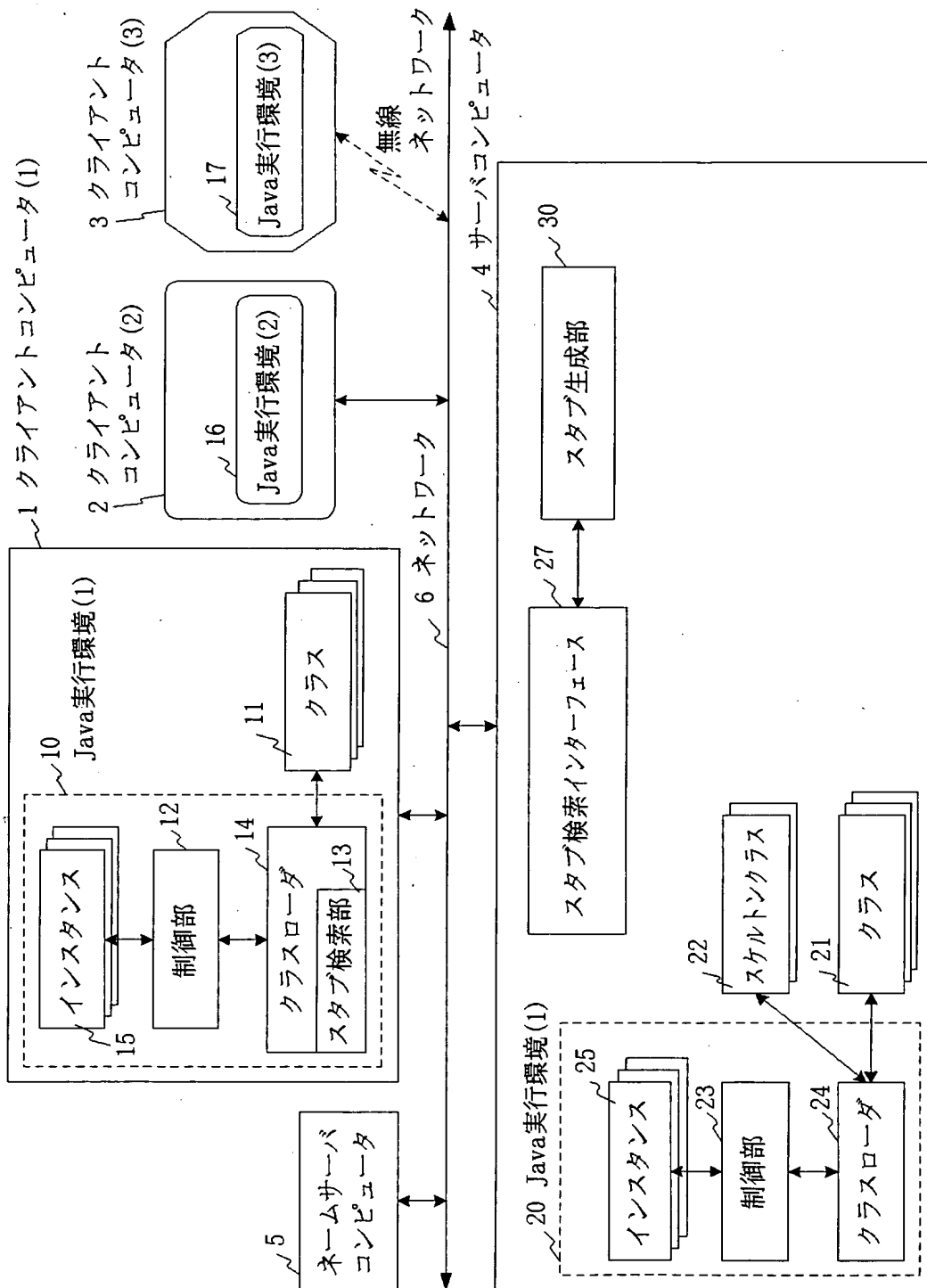
【図6】



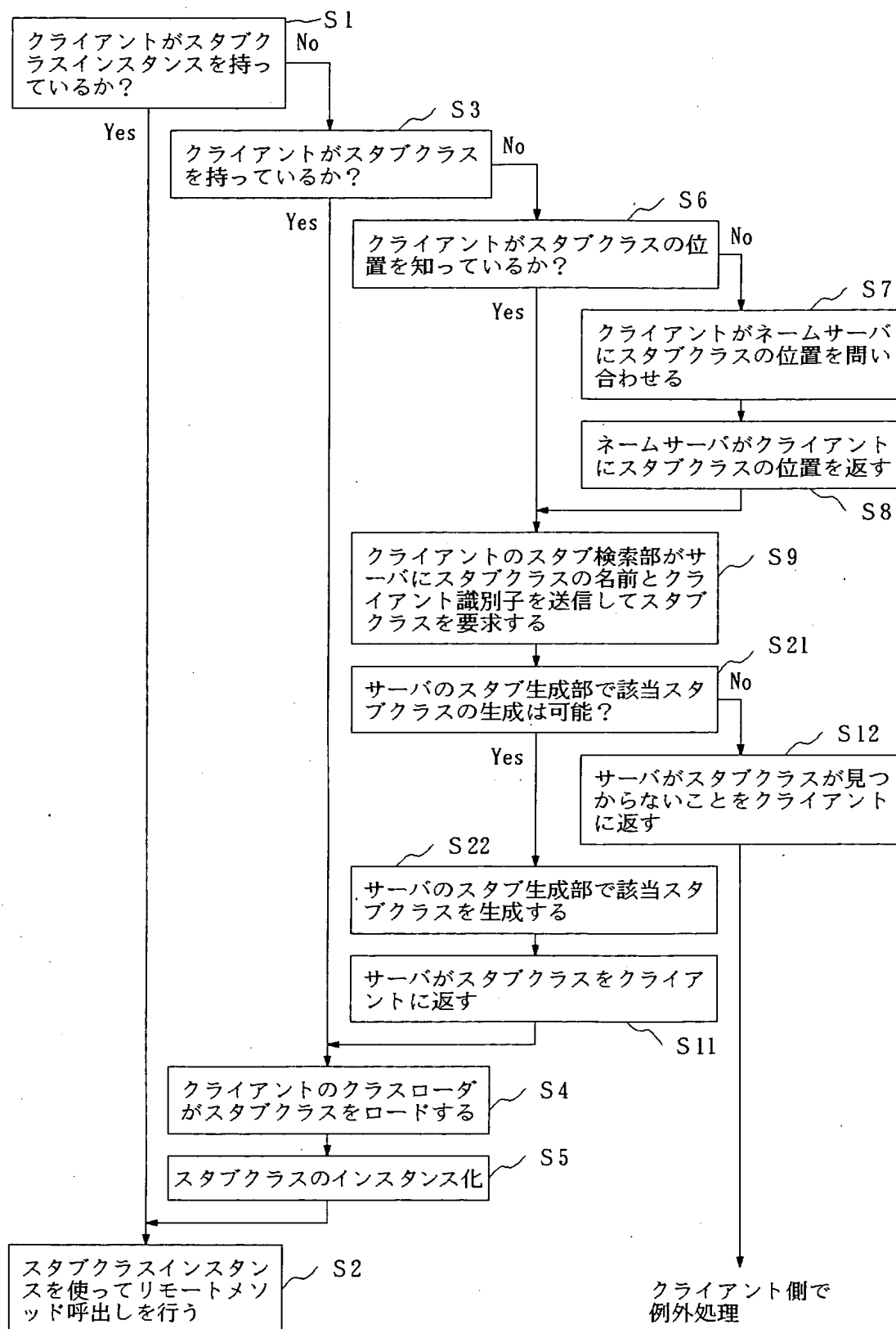
【図7】



【図 8】

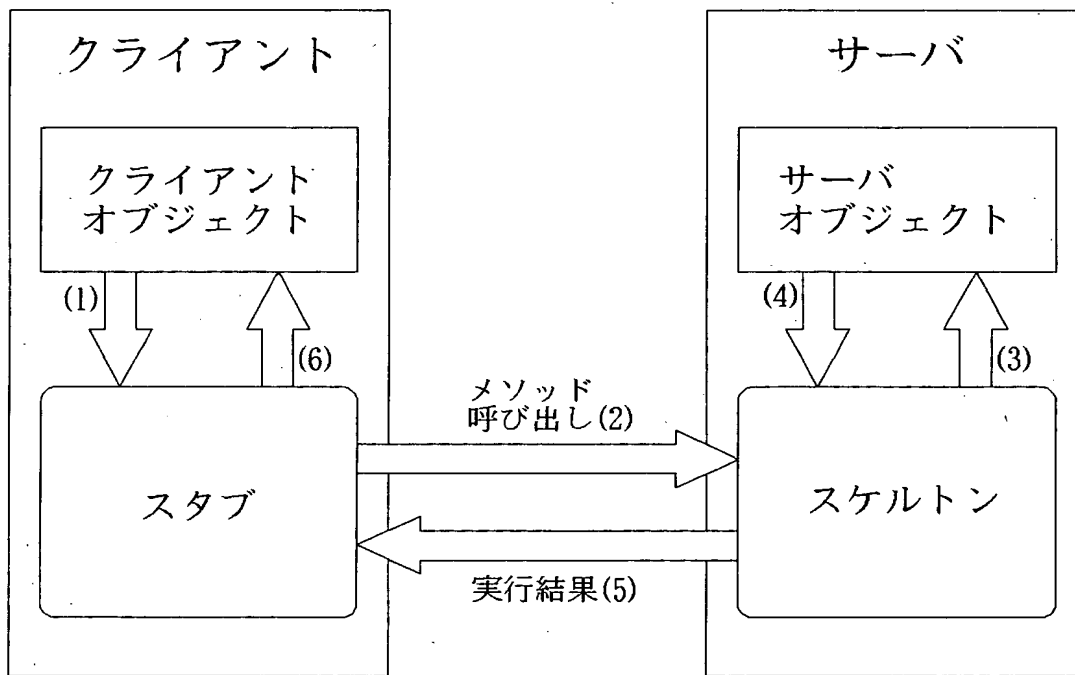


【図 9】

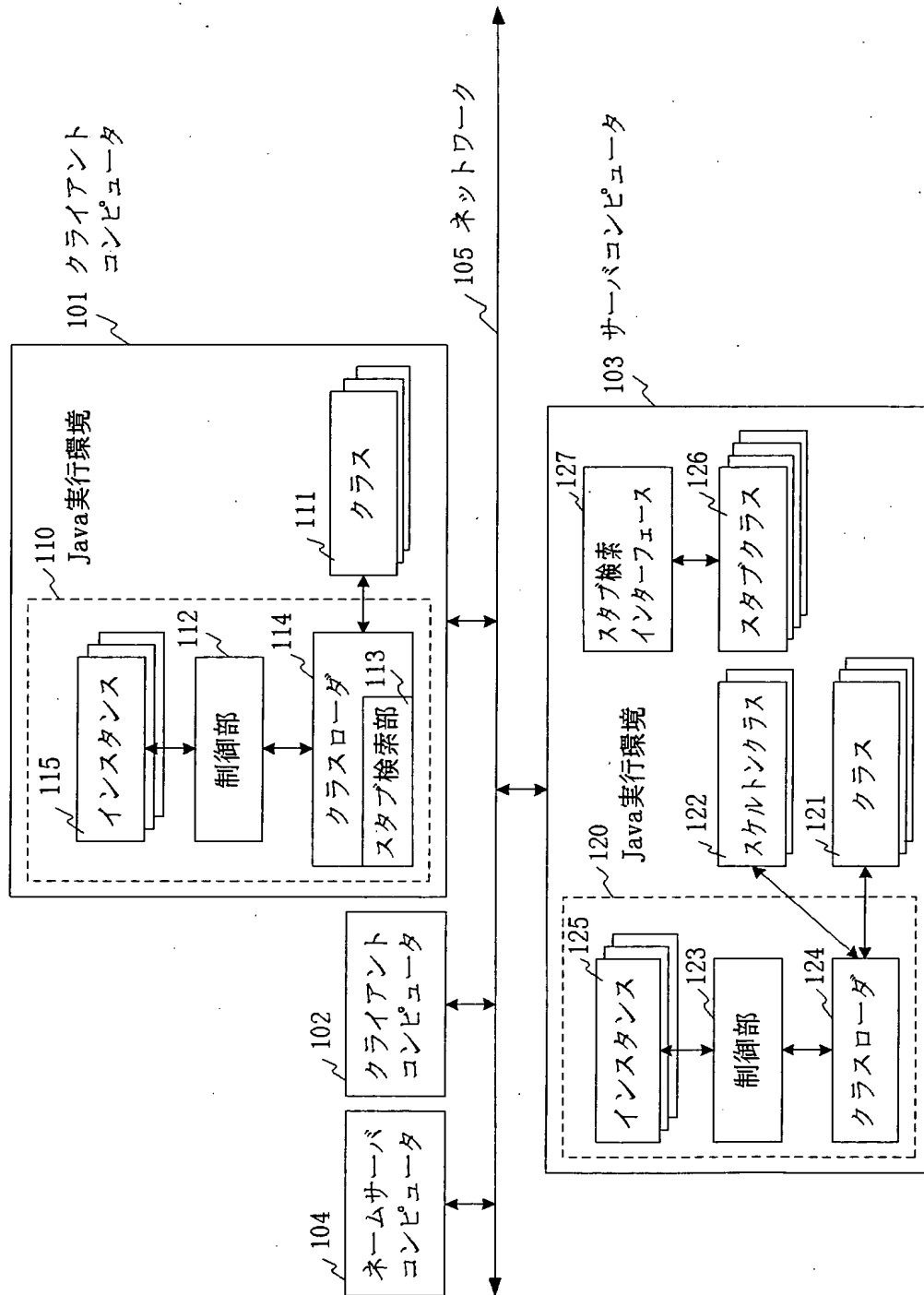




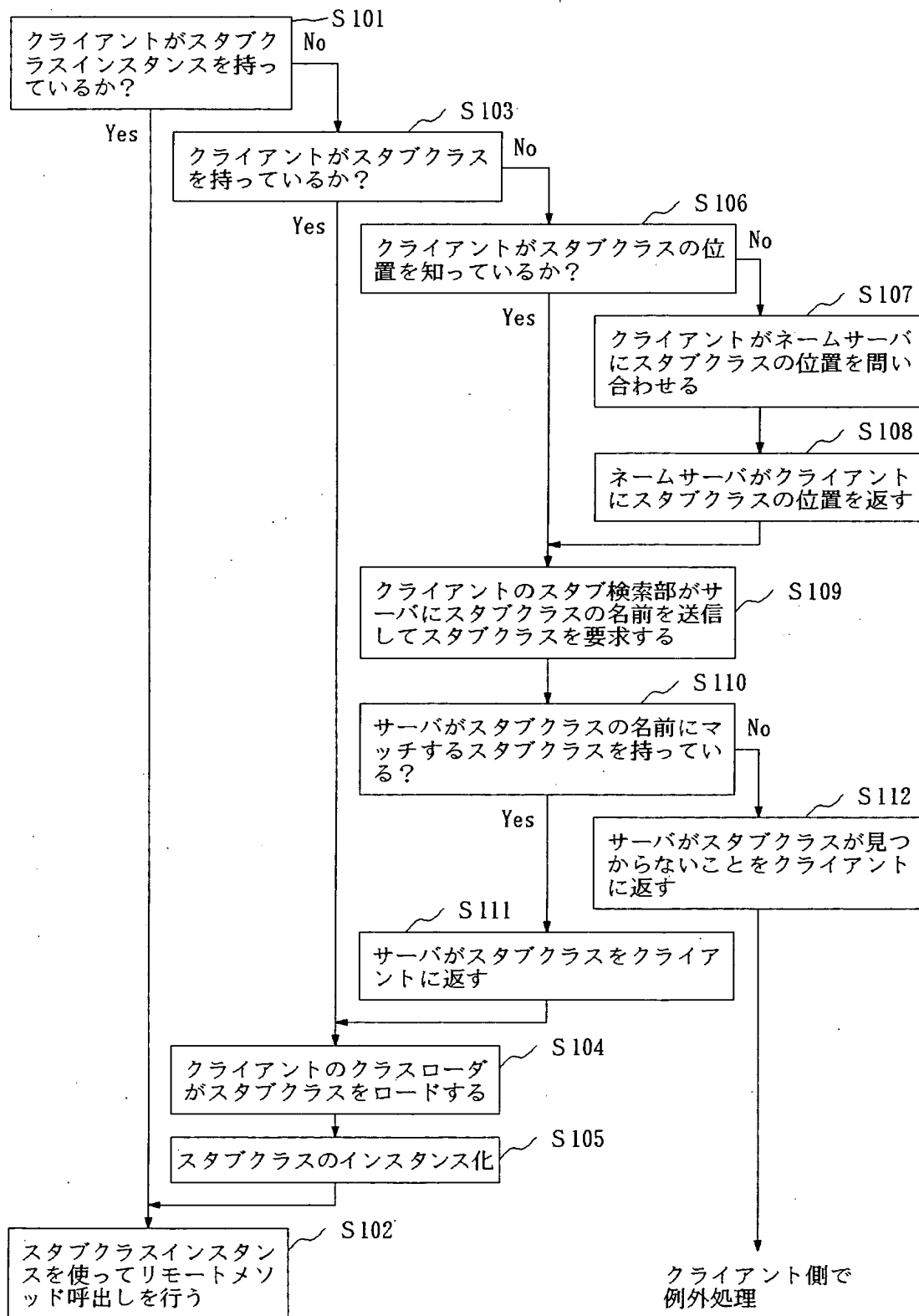
【図10】



【図 11】



【図 12】



【書類名】 要約書

【要約】

【課題】 1つのサーバコンピュータから、J a v a 実行環境の種類が異なる複数のクライアントコンピュータに対して、各クライアントコンピュータにおけるリモートメソッド呼び出しに使えるスタブクラスをダウンロードできるようにする。

【解決手段】 サーバコンピュータ4には、スタブを実行環境の異なるクライアント1～3の種類毎に用意したスタブクラス26-1～26-3がある。各クライアントコンピュータ1～3はスタブが必要になったとき、スタブ名とクライアント識別子とを指定したスタブ要求をネットワーク6経由でサーバコンピュータ4に送出する。サーバコンピュータ4のスタブ検索インターフェース27は、指定されたスタブ名とクライアント識別子に基づいて適切なスタブをスタブクラス26-1～26-3から検索して要求元のクライアントに返す。

【選択図】 図1

出 願 人 履 歴 情 報

識別番号 [000004237]

1. 変更年月日	1990年 8月29日
[変更理由]	新規登録
住 所	東京都港区芝五丁目7番1号
氏 名	日本電気株式会社